Министерство образования и науки Российской Федерации
Санкт-Петербургский Политехнический Университет Петра Великого
—
Институт прикладной математики и механики
Кафедра «Информационная безопасность компьютерных систем»

ЛАБОРАТОРНАЯ РАБОТА № 2

«**Установка и настройка службы каталога Active Directory**»

по дисциплине «Безопасность современных информационных технологий»

Выполнил
студент гр. 33609/3 Елисеев Н.Н.
<подпись>

Преподаватель
Иванов Д.В.
<подпись>

Санкт-Петербург
2018

1 ЦЕЛЬ РАБОТЫ

Изучить способы и механизмы установки и настройки службы каталога Active Directory.

1. Придумать и описать структуру некоторой компании (без привязки к компьютерному обеспечению).

2. Разработать логическую структуру Active Directory, удовлетворяющую структуре компании.

3. Выполнить установку и настройку Active Directory в соответствии с разработанной структурой.

4. Создать следующие административные группы пользователей в Active Directory: Account Managers, Help Desk, Resource Admins, Genereal Admins.

5. Настроить полномочия для созданных групп пользователей:

Account Managers могут только создавать учетные записи для новых сотрудников и удалять их при необходимости.

Help desk могут только разблокировать пользовательские учетные записи в случае их блокировки при неудачном вводе пароля.

Resource Admins должны управлять доступом к информационным ресурсам организации.

General Admins должны обладать полным доступом и контролировать работу администраторов нижних уровней.

6. Создать тестовых пользователей в каждой из групп.

7. Выполнить проверку правильности настройки полномочий путем попыток исполнения соответствующих действий от имен пользователей различных групп.

8. Создать файловый сервер. На файловом сервере должны быть папки для каждого компьютера и каждого пользователя в организации. При входе пользователя на некоторый компьютер, ему должны монтироваться два разделяемых ресурса, соответствующие папке компьютера и папке пользователя на файловом сервере.

9. Создать инструментарий, помогающий в работе администраторам: скрипт для поиска пользователей, у которых скоро истекает срок действия пароля, и скрипт для разблокирования всех учетных записей пользователей в заданном организационном подразделении.

Требования:

• У каждого студента должна быть своя структура организации, повторов быть не может.

• Создание групп пользователей, настройка их полномочий, создание тестовых пользователей и другие настройки должны выполняться автоматизированно с помощью скриптов.

• Логическая структуру Active Directory должна включать в себя, как мимнимум, домены и организационные подразделения (минимум два).

• Демонстрационный стенд должен состоять минимум из трех компьютеров (возможно использование виртуальных машин).

• В состав стенда должны входить сервера на базе Windows 2008 или выше и рабочие станции на базе Windows 7 или выше.

• Количество дней до истечения срока действия пароля для скрипта поиска пользователей, должно являться входным параметром скрипта.

• Название организационного подразделения для скрипта разблокирования учетных записей пользователей должно являться входным параметром скрипта.

2 РЕЗУЛЬТАТЫ РАБОТЫ

В результате работы разработанной логической структуре организации (Рисунок 1) были созданы лес ibks.com и организационные подразделения (OU) – Adminitstration, Teachers, Students, Others. В каждом организационном подразделении были созданы тестовые пользователи.

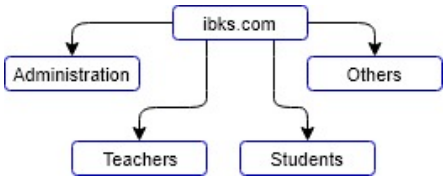

Рисунок 1 –Логическая структура организации

С помощью скриптов были созданы организационные подразделения, а также созданы следующие административные группы: AccountManagers, HelpDesk, ResourceAdmins, GeneralAdmins. На рисунке 2 изображены административные группы в разделе Administration.

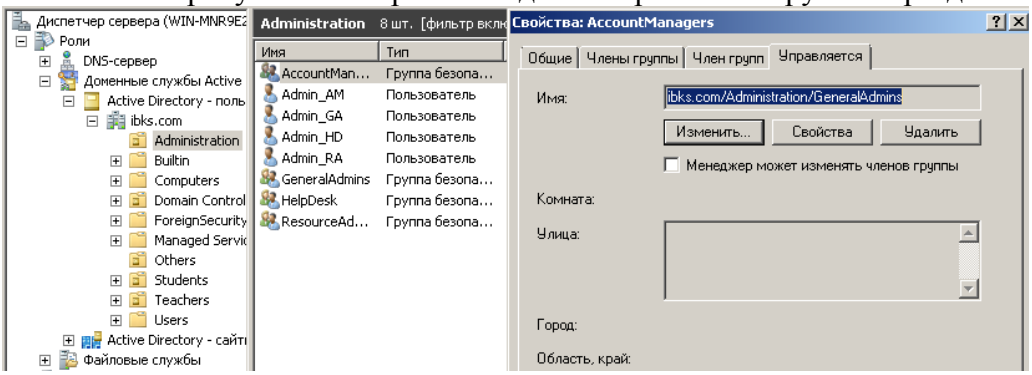

Рисунок 2 – Административные группы и свойства группы AccountManagers

С помощью утилиты командой строки DSACLS.EXE и команды DSMOD GROUP настроены полномочия для созданных групп пользователей.

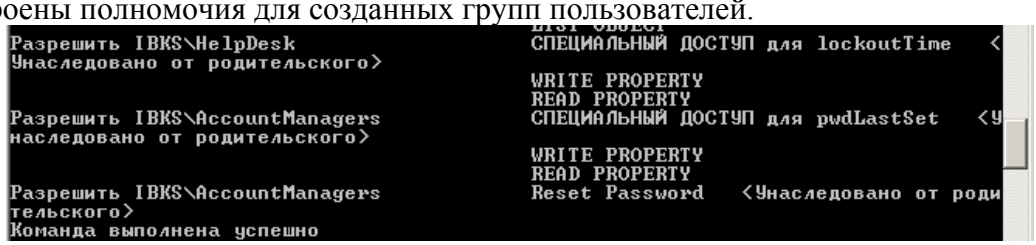

Рисунок 3 – Вывод команды DSACLS.EXE для группы AccountManagers

Чтобы продемонстрировать работоспособность скрипта, которые добавляет и удаляет пользователей, был создан дополнительные пользователь Other2. На рисунках 4, 5 приведен пример запуска скрипта.

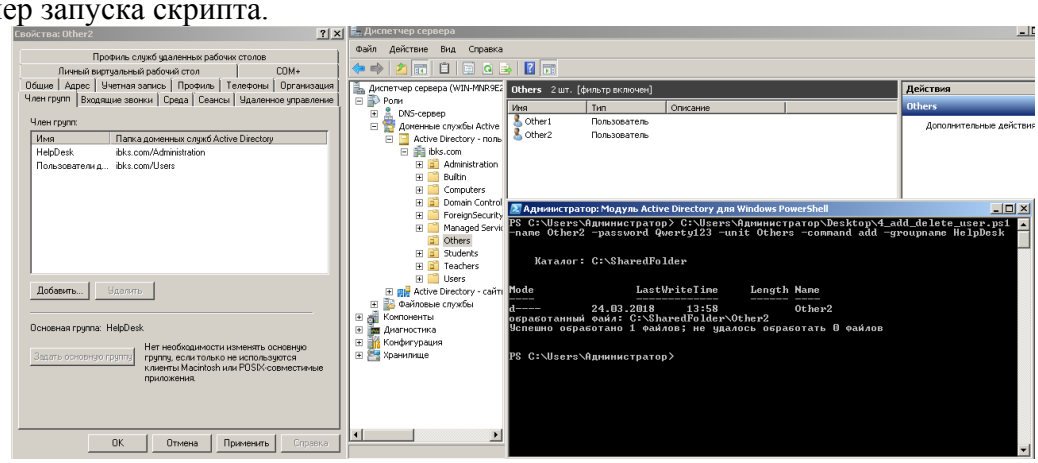

Рисунок 4 – Пример входных параметров скрипта и результат работы добавления пользователя

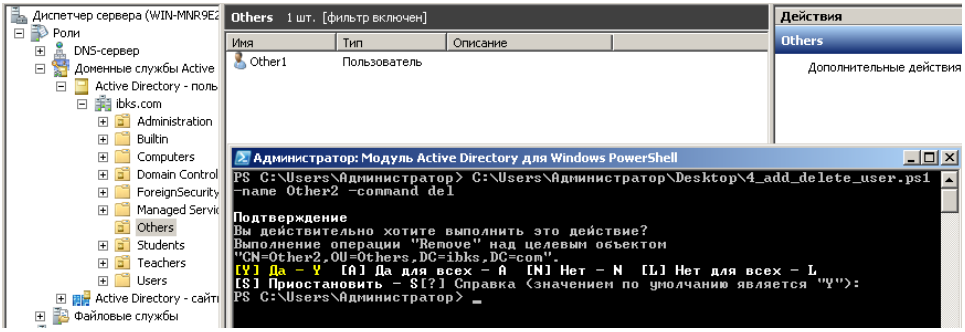

Рисунок 5 – Пример входных параметров скрипта и результат работы удаления пользователя

Одновременно с созданием пользователей, происходит создание каталога с его именем в разделяемом ресурсе. С помощью этого же скрипта, приведённого в приложении, при создании пользователя создаётся каталог, именем которого является имя пользователя, в разделяемом ресурсе – папке Z:\SharedFolder. При входе пользователя в систему ему монтируется диск Z, который ссылается на эту папку. На рисунках 6, 7 приведёны примеры разделения прав доступа и диск в системее для одного из пользователей.

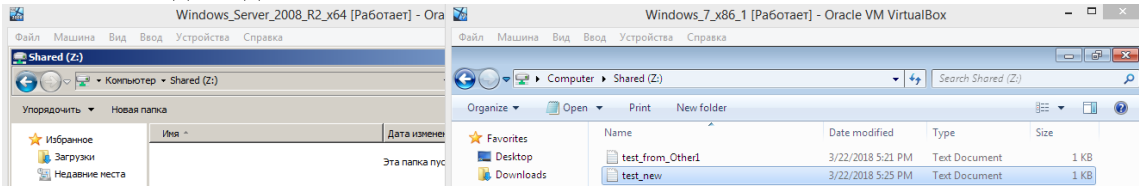

Рисунок 6 – Разделение прав доступа к общей директории

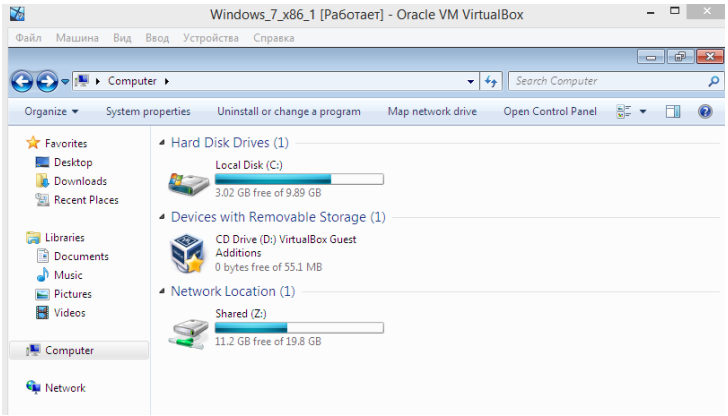

Рисунок 7 – Диск в системе

Были разработаны скрипты для разблокировки пользователй и поиска пользователй с истекающим роком действия пароля.

Рассмотрим скрипт, который позволяет разблокировать пользователей в системе. Политику безопасности Windows можно настроить таким образом, чтобы после некоторого количества неудачных попыток пользователя войти в систему (неверный пароль), его учётная запись была заблокирована. Длительность блокировки также является настраиваемым параметром. Заходим в оснастку Управление групповой политикой – DefaultDomainPolicy – Параметры – Конфигурация компьютера – Политики – Конфигурация Windows – Параметры безопасности – Политики учётных записей/Политика блокировки учётных записей. На рисунке 8 показан, как был настроен параметр ограничения попыток входа в систему.

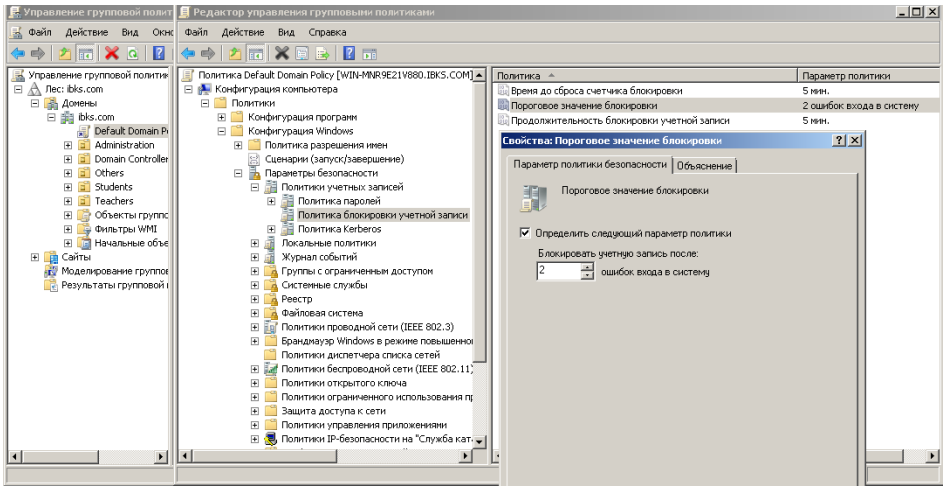

Рисунок 8 – Ограничение попыток входа в систему

На рисунках 9, 10 приведен пример запуска скриптов для разблокироки пользователй и писка пользователей с истекающим сроком действия пароля.

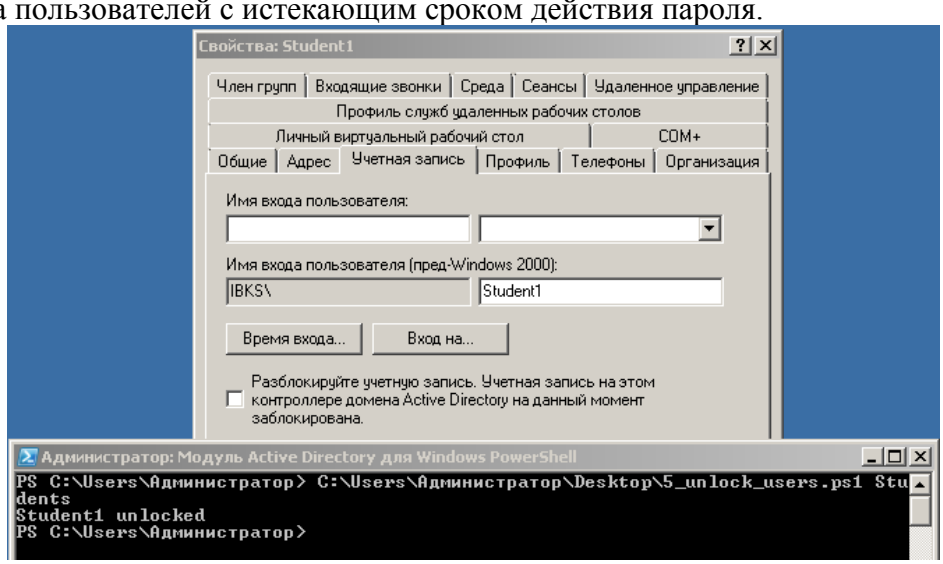

Рисунок 9 – Пример входных параметров скрипта для разблокировки пользователей

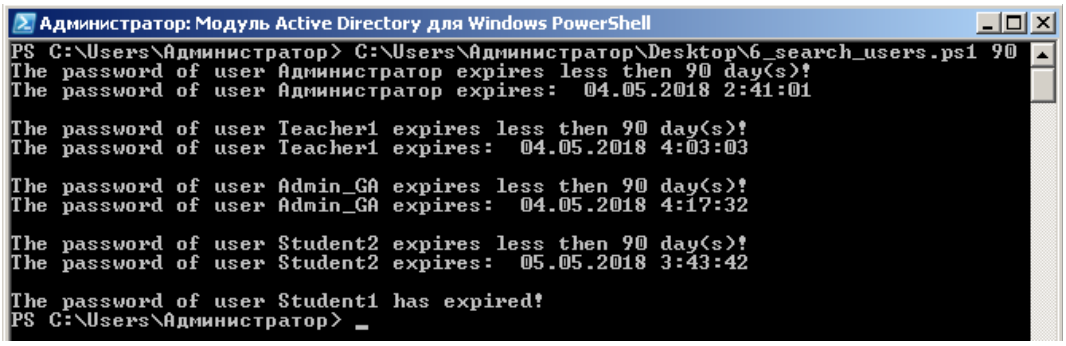

Рисунок 10 – Пример входных параметров скрипта для поиска пользователей с истекающим паролем

3 ВЫВОДЫ

В результате работы были изучены способы и механизмы установки и настройки службы каталога Active Directory. Придумана и описана структура организации, разработана логическая структура Active Directory для данной организации. Созданы необходимые административные группы пользователей.

Листинг 1:

1_create_new_OrganizationUnit.ps1

```
Import-Module ActiveDirectory
$name_dc = "ibks"
$name_dc_com = "com"
# New-ADOrganizationalUnit -Name Others -Path "ou=Department,dc=$name_dc,dc=$name_dc_com"
New-ADOrganizationalUnit -Name Others -Path "dc=$name_dc,dc=$name_dc_com"
New-ADOrganizationalUnit -Name Teachers -Path "dc=$name_dc,dc=$name_dc_com"
New-ADOrganizationalUnit -Name Administration -Path "dc=$name_dc,dc=$name_dc_com"
New-ADOrganizationalUnit -Name Students -Path "dc=$name_dc,dc=$name_dc_com"
```

2_create_new_group.ps1

```
# -groupScope Global - (Action area) Global group
# -ManagedBy "GeneralAdmins" - Контролировать работу администраторов нижних уровней.
Import-Module ActiveDirectory
$name_ou = "Administration"
$name_dc = "ibks"
$name_dc_com = "com"
New-ADGroup -Name "GeneralAdmins" -groupScope Global -Path "ou=$name_ou,dc=$name_dc,dc=$name_dc_com"
New-ADGroup -Name "AccountManagers" -groupScope Global -ManagedBy "GeneralAdmins" -Path
"ou=$name_ou,dc=$name_dc,dc=$name_dc_com"
New-ADGroup -Name "HelpDesk" -groupScope Global -ManagedBy "GeneralAdmins" -Path
"ou=$name_ou,dc=$name_dc,dc=$name_dc_com"
New-ADGroup -Name "ResourceAdmins" -groupScope Global -ManagedBy "GeneralAdmins" -Path
"ou=$name_ou,dc=$name_dc,dc=$name_dc_com"
```

3_rules_for_manager.bat

```
REM CN = Common Name; OU = Organizational Unit; DC = Domain Component
REM /I - параметр для указания флага наследования
REM :T - наследуемые права доступа распространяются к этому объекту и подобъектам
REM :S - наследуемые права доступа распространяются только на подобъекты
REM /G - группа
REM CC - создать дочерний объект
REM ;user - только пользовательские объекты
REM GR - Generic Read, GW - Generic Write
REM CA - Control access right
REM RP - Read property, RW - Write property
REM pwdLastSet - поменять пароль при след. входе
REM lockoutTime - может разблокировать пользователя
REM Утилита командной строки DSACLS.EXE позволяет просматривать и
REM изменять права доступа пользователей и групп пользователей
dsacls.exe ou=Administration,dc=ibks,dc=com /I:T /G "ibks\AccountManagers":CC;user
dsacls.exe ou=Administration,dc=ibks,dc=com /I:S /G "ibks\AccountManagers":GRGW;;user
dsacls.exe ou=Administration,dc=ibks,dc=com /I:S /G "ibks\AccountManagers":CA;"Reset Password";user
dsacls.exe ou=Administration,dc=ibks,dc=com /I:S /G "ibks\AccountManagers":RPWP;pwdLastSet;user
dsacls.exe ou=Administration,dc=ibks,dc=com /I:S /G "ibks\HelpDesk":RPWP;lockoutTime;user
REM dsmod group - Изменяет атрибуты одной или нескольких существующих групп в каталоге
REM -addmbr - Указывает, что члены группы, заданные параметром, добавляются к группе
REM IIS_IUSRS - Встроенная группа, которую используют службы IIS (Internet Information Services)
REM Builtin - Встроенные группы
dsmod group "cn=IIS_IUSRS,cn=Builtin,dc=ibks,dc=com" -addmbr "cn=ResourceAdmins,ou=Administration,dc=ibks,dc=com"
REM Если Windows английская cn = Administrators
REM dsmod group "cn=Administrators,cn=Builtin,dc=ibks,dc=com" -addmbr "cn=GeneralAdmins,ou=Administration,dc=ibks,dc=com"
REM Если используются русские имена, то необходимо сделать кодировку OEM 866, или вбейте команду руками
REM Если Windows русская cn = Администраторы
dsmod group "cn=Администраторы,cn=Builtin,dc=ibks,dc=com" -addmbr "cn=GeneralAdmins,ou=Administration,dc=ibks,dc=com"
```

4_add_delete_user.ps1

```
param(
   [string]$name = $(throw "Enter name"),
   [string]$password = "Qwerty12",
   [string]$unit = "Students",
   [string]$command = $(throw "add or del"),
   [string]$groupname = ""
)
switch($command)
{
        add
        {
                if ($unit)
                {
                        $secpass = $password | ConvertTo-SecureString -AsPlainText -Force
                        # делим строку на подстроки, разделенные '/'
                        $outmp = $unit -split '/'
                        # строим путь в подраздел
                        $outmp | ForEach-Object {$path = "OU=$_," + $path}
                        # в конце добавляем домен
```

```
                        $path +=  "DC=ibks, DC=com"
                        #-Name $name ` - имя пользователя
                        #-Path $path ` - путь до подраздела
                        #-SamAccountName $name ` - имя учетной записи
                        #-DisplayName $name ` - отображаемое имя
                        #-AccountPassword $secpass ` - пароль
                        #-ChangePasswordAtLogon $true ` - сменить пароль при первом входе
                        #-Enabled $true ` - включить учетный запист
                        #-HomeDrive "Z:" ` -
                        #-HomeDirectory "\\WIN-MNR9E21V880\ImShared\$name" -
                        New-ADUser `
                                -Name $name `
                                -Path $path `
                                -SamAccountName $name `
                                -DisplayName $name `
                                -AccountPassword $secpass `
                                -ChangePasswordAtLogon $true `
                                -Enabled $true `
                                -HomeDrive "Z:" `
                                -HomeDirectory "\\IBKS\ImShared\$name"
                        mkdir "C:\SharedFolder\$name"
                        # iCACLS отображает или изменяет списки управления доступом (Access Control Lists (ACLs))
                        # /grant[:r] SID:permission — предоставление указанных прав доступа пользователя
                        # Права наследования могут предшествовать любой форме и применяются только к каталогам:
                                # (OI) - наследование объектами
                                # (CI) - наследование контейнерами
                        # M[edium]: средний
                        # Для данного пользователся даем доступ к его папке с правами наследоваия и "средний" доступ
                        icacls "C:\SharedFolder\$name" /grant "$($name):(OI)(CI)M"
                        if ($groupname)
                        {
                                Add-ADGroupMember -Identity $groupname -Members $name #$user
                        }
                }
        }
        del
        {
                Remove-ADUser -Identity $name
                Remove-Item "C:\SharedFolder\$name"
        }
}
```

5_unlock_users.ps1

```
param([string]$unit = $(throw "Enter organization unit"))
if ($unit)
{
   # делим строку на подстроки, разделенные '/'
   $outmp = $unit -split '/'
   # строим путь в подраздел
   $outmp | ForEach-Object {$path = "OU=$_," + $path}
   # в конце добавляем домен
   $path += "DC=ibks, DC=com"
}
# -UsersOnly - только пользователи
# -LockedOut - заблокированные
# -SearchBase - поиск в заданном подразделении
# Search-ADAccount возвращает список учетных записей
$list = Search-ADAccount -SearchBase $path -UsersOnly -LockedOut

if ($list)
{
        foreach ($user in $list)
        {
           # разблокировка учетной записи по имени
           Unlock-ADAccount -Identity $($user.Name)

           Write-Host $user.Name "unlocked"
        }
}
else
{
   Write-Host No find locked users
}
```

6_search_users.ps1

```
# Скрипт ищет во всем домене пользователей с истекающим сроком действия пароля
# Входной параметр - количество дней до истечения срока действия
param([int]$number = 5)
# получаем список пользователей всего домена,
# которые включены и имеют срок действия пароля
```

```
# PasswordNeverExpires       - пароль без срока действия
#$users_expire_date = &Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} `
# –Properties - какие атрибуты пользователя нужно выводить
# SamAccountName - Имя входа пользователя
# msDS-UserPasswordExpiryTimeComputed - Содержит время истечения срока действия текущего пароля пользователя
# PasswordLastSet - Когда последний раз пользователь менял пароль
# Знак '|' в конце - передаем список в Select-Object (в следующий команду)
#–Properties "SamAccountName", "msDS-UserPasswordExpiryTimeComputed", "PasswordLastSet" | `
# Выбираем объект из списка с указанными свойствами
# @{...} - ассоциативный массив
# Массив состоит из имени и времени истечения срока действия пароля
#Select-Object –Property “SamAccountName",@{Name="ExpiryDate";`
# Получаем время истечения пароля в виде строки и конвертируем
# его в datetime FromFileTime - форматирование в виде MM/dd/yyyy
#Expression={[datetime]::FromFileTime($_."msDS-UserPasswordExpiryTimeComputed")}}
$users_expire_date = &Get-ADUser -filter {Enabled -eq $True -and PasswordNeverExpires -eq $False} `
–Properties "SamAccountName", "msDS-UserPasswordExpiryTimeComputed", "PasswordLastSet" | `
Select-Object –Property “SamAccountName",@{Name="ExpiryDate";`
Expression={[datetime]::FromFileTime($_."msDS-UserPasswordExpiryTimeComputed")}}
# проходимся по каждому пользователю из списка
foreach($item in $users_expire_date)
{
   # получаем дату истечения действия пароля в US
   $expire_date = $item.ExpiryDate
   # получаем текущую дату в формате US
   $current_date = &Get-Date
   # получаем имя пользователя для входа
   $DisplayName = $item.SamAccountName
   # -gt (Great than) - если больше
   if($current_date -gt $item.ExpiryDate)
   {
     Write-Host "The password of user" $DisplayName "has expired!"
     Exit
   }
   # получаем время до истечения действия пароля
   $lasts = $expire_date - $current_date
   # -lt (Less than) - если меньше
   if($lasts.Days -lt $number)
   {
      Write-Host "The password of user" $DisplayName "expires less then" $Number "day(s)!"
      Write-Host "The password of user" $DisplayName "expires: " $expire_date
      Write-Host " "
   }
}
```