


# Скрипты для просмотра сведений о сертификате в атрибуте msDS-KeyCredentialLink - Windows Server

 [learn.microsoft.com/ru-ru/troubleshoot/windows-server/support-tools/script-to-view-msds-keycredentiallink-attribute-value](https://learn.microsoft.com/ru-ru/troubleshoot/windows-server/support-tools/script-to-view-msds-keycredentiallink-attribute-value)

## Скрипты. Просмотр сведений о сертификате в атрибуте msDS-KeyCredentialLink из пользовательских объектов AD

- Статья
- 13.02.2025

Атрибут **msDS-KeyCredentialLink** можно просмотреть в PowerShell. Однако значение находится в двоичном формате и не может быть прочитано. Этот скрипт помогает выполнить следующие действия:

- **msDS-KeyCredentialLink**.
- Извлеките хэш ключа bcrypt-sha256 каждого сертификата, сохраненного в **msDS-KeyCredentialLink**, и сохраните его в файле.

Затем можно использовать сохраненные сведения, чтобы проверить, находятся ли ожидаемые значения в атрибуте **msDS-KeyCredentialLink** **пользователя**.

## Script

Важно!

Этот пример скрипта не поддерживается в любой стандартной программе или службе поддержки Майкрософт.

Пример сценария приводится в виде "как есть", без каких-либо гарантий. Кроме того, корпорация Microsoft отказывается от всех подразумеваемых гарантий, включая в том числе все подразумеваемые гарантии пригодности для продажи или определенной цели.

Весь риск, возникающий из использования или производительности примеров скриптов и документации, остается с вами. Корпорация Майкрософт, ее штатные авторы и другие лица, принимающие участие в создании, подготовке и выпуске сценариев, ни при каких обстоятельствах не несут ответственность за какой-либо ущерб (в том числе, ущерб, вызванный потерей доходов предприятия, остановкой его работы, потерей бизнес-данных и другими материальными потерями), вызванный использованием или неспособностью использовать примеры сценариев и документацию, даже если корпорации Майкрософт известно о возможности нанесения такого ущерба.

После запуска скрипта результаты сохраняются в файле  
**C:\temp\KeyCredentialLink-report.txt**.



```

$outputfile = "C:\temp\KeyCredentialLink-report.txt"
New-Item -ItemType file -Path $outputfile -Force | Out-Null

"Report generated on " + (Get-Date) | Out-File $outputfile

# Enumerate all AD users that has a msds-KeyCredentialLink value
foreach ($user in (Get-ADUser -LDAPFilter '(msDS-KeyCredentialLink=*)' -Properties
"msDS-KeyCredentialLink")) {

    # For each user, output the UPN, DN, and all key IDs in msDS-KeyCredentialLink
    "=====`nUser: $($user.UserPrincipalName)`nDN:
$($user.DistinguishedName)" | Out-File $outputfile -Append
    "KeyCredialLink Entries:" | Out-File $outputfile -Append
    "    Source|Usage|DeviceID                                |KeyID" | Out-File
$outputfile -Append
    "    -----" |
Out-File $outputfile -Append

    foreach ($blob in ($user."msDS-KeyCredentialLink")) {
        $KCLstring = ($blob -split ':')[2]

        # Check that the entries are version 2
        if ($KCLstring.Substring(0, 8) -eq "00020000") {
            $currentIndex = 8

            # Parse all KeyCredentialLink entries from the hex string
            while ($currentIndex -lt $KCLstring.Length) {

                # Read the length, reverse the byte order to account for
endianess, then convert to an int
                # The length is in bytes, so multiply by 2 to get the length in
characters
                $strLength = ($KCLstring.Substring($currentIndex, 4)) -split '(?
<=\\G..)(?!$)'
                [array]::Reverse($strLength)
                $kcle_Length = ([convert]::ToInt16(-join $strLength, 16)) * 2

                # Read the identifier and value
                $kcle_Identifier = $KCLstring.Substring($currentIndex + 4, 2)
                $kcle_Value = $KCLstring.Substring($currentIndex + 6, $kcle_Length)

                switch ($kcle_Identifier) {
                    # KeyID
                    '01' {
                        $KeyID = $kcle_Value
                    }

                    # KeyUsage
                    '04' {
                        switch ($kcle_Value) {
                            '01' { $Usage = "NGC  " }
                            '07' { $Usage = "FIDO  " }
                            '08' { $Usage = "FEK  " }
                            Default { $Usage = $kcle_Value }
                        }
                    }
                }

                $currentIndex += $kcle_Length + 2
            }
        }
    }
}

```

```

# Source
'05' {
    switch ($kcle_Value) {
        '00' { $Source = }
        '01' { $Source = "Entra " }
        Default { $Source = $kcle_Value }
    }
}

# DeviceID
'06' {
    $tempByteArray = $kcle_Value -split '(?<=\G..)(?!$)'
    $DeviceID = [System.Guid]::new($tempByteArray[3..0] +
    $tempByteArray[5..4] + $tempByteArray[7..6] + $tempByteArray[8..16] -join "")
}

$curIndex += 6 + $kcle_Length
}

# Save the data to file
" $Source|$Usage|$DeviceID|$KeyID" | Out-File $outputfile -Append
}
}
}

```

## Пример выходных данных и анализа скрипта

Ниже приведен пример выходных данных скрипта:

### Выходные данные

```

User: user1@contoso.com
DN: CN=user1,OU=MyOU,DC=contoso,DC=com
KeyCredentialLink Entries:
    Source|Usage|DeviceID                                     | KeyID
    -----
    Entra |NGC   |8a763ab0-0f6f-44f3-99ae-
599a6aaca45b|FD68391824C44158B23C5605F567A588D02C4B2962AC96B789EDBCE091CF5067
    Entra |NGC   |9cf88f41-1e1e-462b-87d5-
6938410ea82c|E91EF4E058513155A3F7E7E5B3E34951ADE923FFD0A7C24BB9957510F007E2F3
    Entra |NGC   |d04581fc-d1c8-45fc-a5ad-
192bc649574f|E60B476088CC5CDF6FB75454CFA6E17C3059D51F2CA1E414E1554715BE6C0527
=====
User: user2@contoso.com
DN: CN=user2,OU=MyOU,DC=contoso,DC=com
KeyCredentialLink Entries:
    Source|Usage|DeviceID                                     | KeyID
    -----
    Entra |NGC   |c8fcc7a6-8f3f-4ec7-a90b-
49c6988ba3a4|32EF67B902CB498710F0091F5B10B6A4A2F05D621B748B8150E08FA3048F227F

```

Каждая **KeyCredentialLink** запись представляет сертификат. Выходные данные содержат следующие сведения:

- **Source**: источник сертификата. Эти сведения могут быть из идентификатора Microsoft Entra или локальной службы AD.
- **Usage**: определенное использование сертификата. Эти сведения могут быть NGC (WHfB), FIDO или FEK (ключ шифрования файлов).
- **DeviceID**: идентификатор компьютера, на котором был создан сертификат. Это идентификатор устройства в идентификаторе Microsoft Entra и objectGUID в AD.
- **KeyID**: хэш ключа bcrypt-sha256 сертификата.

Соответствующий сертификат должен находиться в личном хранилище сертификатов пользователя на компьютере с соответствующим **DeviceID**. Чтобы найти сертификат, используемый на клиенте, можно запустить `certutil -v -user -store my` из PowerShell или командной строки, чтобы дампа подробные сведения о сертификате из личного хранилища пользователя. В этом примере необходимо найти самозаверяющий сертификат, в котором субъект и издатель одинаковы и в форме `CN=<User SID>/login.windows.net/<Tenant ID>/<user UPN>`. После поиска этого сертификата проверьте хэш ключа bcrypt-sha256. Это хэш-значение должно соответствовать одному из записей в атрибуте **msDS-KeyCredentialLink**.

Вот фрагмент из user2@contoso.com хранилища сертификатов:

Выходные данные

```

> certutil -v -user -store my
my "Personal"
===== Certificate 0 =====
X509 Certificate:
Version: 3
Serial Number: 184621...
Signature Algorithm:
    Algorithm ObjectId: 1.2.840.113549.1.1.11 sha256RSA
    Algorithm Parameters:
        05 00
Issuer:
    CN=S-1-5-21-394...7436/login.windows.net/ccf...83a/user2@contoso.com

:

Signature Algorithm:
    Algorithm ObjectId: 1.2.840.113549.1.1.11 sha256RSA
    Algorithm Parameters:
        05 00
Signature: UnusedBits=0
    0000 19 17 8f 65 c1 e3 f1 0a 3b 62 90 7f fa 94 13 ad
// snip
    00f0 a2 e3 72 54 a5 0a 84 30 9e 8b 81 19 d3 61 46 58
Signature matches Public Key
Root Certificate: Subject matches Issuer
Key Id Hash(rfc-sha1): 9a546...
Key Id Hash(sha1): d66eb...
Key Id Hash(bcrypt-sha1): 7f4a0...
Key Id Hash(bcrypt-sha256): 32ef67b902cb...

```

## Справка

---

Структуры ссылок на ключевые учетные данные