

Storing the BitLocker volume keys using a Custom Field

Use a FileWave Custom Field to store the volume keys for your BitLocker volumes. This can be helpful if you don't have another way to escrow the volume keys. The Custom Field outlined in this article will get the volume key for every volume so if there is an encrypted C: and D: you would see both reported by this field.

Adding the Custom Field

1. Download the following Custom Field export: [BitLocker Key Custom Field.customfields](#)
2. Import the downloaded file into "FileWave Admin>Assistants>Custom Fields>Edit Custom Fields>Import".
3. Save changes within Custom Fields dialog.
4. Associate Custom Field with desired Windows devices via "right-click>Edit Custom Field(s) Associations".
 1. A Windows-based Smart Group is very helpful to quickly associate Custom Field
 2. Smart Group criteria: "Client OS Platform [equals] Windows"

The screenshot shows the 'Custom Fields' dialog in FileWave Admin. On the left, a list of custom fields is shown, with 'BitLocker Key' selected. The right pane shows the 'Field Details' for 'BitLocker Key'. The 'Name' is 'BitLocker Key' and the 'Internal Name' is 'bitlocker_key'. The 'Description' is empty. The 'Provided By' is 'Client Script'. The 'Assigned to all devices' checkbox is checked. The 'Data Type' is 'String'. The 'Client Script' section shows a PowerShell script that retrieves BitLocker volume keys. The script is as follows:

```
# FileWave client will execute this script. The output will be used as the value of the custom field.
#
# Below is an example of how to read the value of one ENVIRONMENT VARIABLE in your script:

# $my_var = $Env:ENV_VAR_NAME
#
# Identify all the Bitlocker volumes.
$BitlockerVolumers = Get-BitLockerVolume

# For each volume, get the RecoveryPassowrd and display it.
$BitlockerVolumers |
ForEach-Object {
    $MountPoint = $_.MountPoint
    $RecoveryKey = [string](($_.KeyProtector).RecoveryPassword
    if ($RecoveryKey.Length -gt 5) {
        Write-Output ("{$MountPoint,$RecoveryKey}")
    }
}
```

Here is the script from the Custom Field:

```
# FileWave client will execute this script. The output will be used as the value of the custom field.
#
# Below is an example of how to read the value of one ENVIRONMENT VARIABLE in your script:

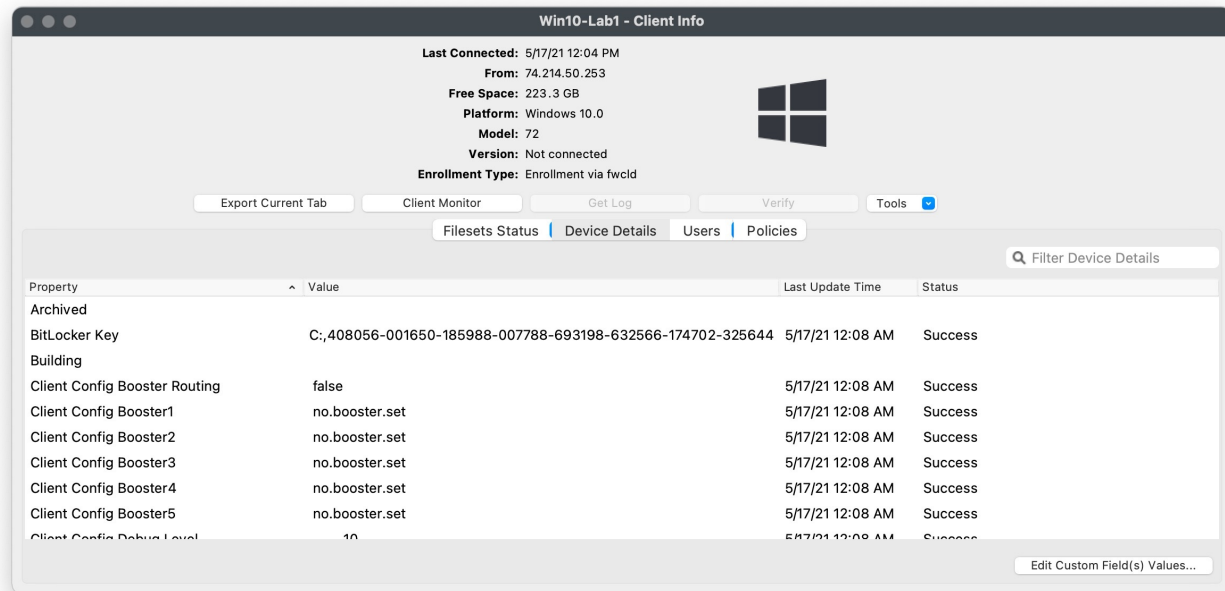
# $my_var = $Env:ENV_VAR_NAME
#
# Identify all the Bitlocker volumes.
$BitlockerVolumers = Get-BitLockerVolume

# For each volume, get the RecoveryPassowrd and display it.
$BitlockerVolumers |
    ForEach-Object {
        $MountPoint = $_.MountPoint
        $RecoveryKey = [string](($_.KeyProtector).RecoveryPassword
        if ($RecoveryKey.Length -gt 5) {
            Write-Output ("{$MountPoint,$RecoveryKey}")
        }
    }
}
```

Assigning the Custom Field to devices

1. Save changes within Custom Fields dialog.
2. Associate Custom Field with desired Windows devices via "right-click>Edit Custom Field(s) Associations".
 - A Windows-based Smart Group is very helpful to quickly associate Custom Field
 - Smart Group criteria: "Client OS Platform [equals] Windows"
3. Alternatively you could assign the field to all devices since only Windows devices will run the script.

Results



Related articles

- [Securing FileWave Server on the Internet for Remote Device Management](#)

🔄Revision #2

★Created 13 July 2023 20:18:34 by Josh Levitsky

✎Updated 16 July 2023 23:56:30 by Josh Levitsky