STOPPING LATERAL MOVEMENT BITLOCKER ATTACKS (WITH SOME SYSTEM HARDENING POWERSHELL AT THE END)

BitLocker, a built-in Windows encryption solution, is widely trusted for endpoint protection. But attackers continue to evolve and recent research highlights sophisticated new techniques for bypassing BitLocker's defenses. Here's how these attacks work, and what you can do to stay one step ahead.

How Attackers Are Exploiting BitLocker

1. BitlockMove: COM/DCOM Hijacking and Lateral Movement

Malicious actors have found ways to exploit BitLocker's COM classes when configured for the "INTERACTIVE USER" session. By deploying a malicious COM hijack and dropping a DLL via SMB, attackers can force a Windows system to load and execute code within a logged-on user's session; even remotely via DCOM.

Why does this matter?
This method enables remote code execution under the user's security context, no stolen credentials, LSASS dumping, or impersonation required. This attack can come from bad downloads, website drops, just frankly "the internet."

2. Bitpixie: Bootloader Memory Extraction Attack

A second, equally concerning vector involves memory extraction attacks during the boot process. Here, an attacker triggers a soft reboot using PXE and custom bootloader manipulation (such as booting into Windows Recovery or PE). This technique leaves the **Volume Master Key (VMK)** in system memory, which attackers can then extract using a Linux kernel module or Windows PE tool thus bypassing BitLocker entirely.

Who's most at risk?
Systems relying solely on TPM for BitLocker (with no pre-boot PIN or USB key) are particularly vulnerable.

How to Defend Against These Attacks

A. Protecting Against BitlockMove (COM/DCOM Hijacking)

- **Lock Down COM Registrations:**
  Use AppLocker or Group Policy to prevent unauthorized COM class registrations or hijacking in interactive sessions.
- **Harden SMB Sharing:** Restrict SMB write access to sensitive registry paths and enforce least privilege on shares.
- **Implement Endpoint Protection Rules:** Deploy hardening policies or EDR solutions that monitor and block suspicious COM DLL loads; especially targeting BitLocker-related CLSIDs.
- **Session Isolation:** Minimize exposure of interactive sessions. Disable COM activation for BitLocker classes if not needed.

B. Mitigating Bitpixie and Bootloader Memory Exposure

- **Enable Pre-Boot Authentication:** Require a PIN or USB key in addition to TPM, so that the VMK isn't loaded without user intervention.
- **Disable PXE Boot:** Turn off network/PXE boot options in BIOS/UEFI unless absolutely necessary.
- **Apply Microsoft Updates & Secure Boot Enhancements:** Stay up to date with patches and Secure Boot certificate updates to prevent bootloader downgrades or bypasses.
- **Monitor TPM PCR Policies:** Enforce stricter Platform Configuration Register policies to detect unauthorized bootloader changes.
- **Disable DMA Ports or Enable Kernel DMA Protection:** Block physical memory access through Thunderbolt, PCIe, FireWire, and enable Virtualization-Based Security (VBS) where supported.

C. Additional General Hardening Measures

- **Require Clean Shutdowns:**
  Cold boot and memory remanence attacks are best mitigated by shutting down devices…not just putting them to sleep or hibernating, if its not going to be used for more than a lunch break; shut it down.
- **Use VBS and IOMMU:**
  Virtualize sensitive memory regions and block unauthorized DMA access.
- **Limit Physical Access:**
  Restrict the use of physical ports and prevent unauthorized bootable media.
- **Prioritize Backups:**
  Always keep offline or offsite backups. Ransomware variants (such as ShrinkLocker) can weaponize BitLocker itself to lock you out, so having reliable backups is critical.

PowerShell Scripts for Real-World Defense

Here are some PowerShell Scripts that I created to assist in ensuring that your system is hardened to these types of attacks. Feel free to use them, just please give dues where they are deserved… Don't claim these are your work.

1. First you need to ensure that you have admin access to your device and that you have snapshotted the device (I am not responsible if you make your PC or PC's go Boom Boom)
2. Run the following PowerShell Script as Administrator

```
# Written by N4Vx0.2
# This script enumerates all COM CLSIDs in HKLM:\SOFTWARE\Classes\CLSID
# Filters for BitLocker-related COM and checks for "INTERACTIVE USER" permissions

$bitlockerKeywords = @("bitlocker", "fve", "encrypt", "decryption") # Adjust keywords as needed

$clsidRoot = "HKLM:\SOFTWARE\Classes\CLSID"
$bitlockerClsids = @()

# Enumerate all CLSIDs
Get-ChildItem $clsidRoot | ForEach-Object {
   $clsidKey = $_.PsPath
   $defaultValue = (Get-ItemProperty -Path $clsidKey -Name '(default)' -ErrorAction SilentlyContinue)."(default)"

   # Search for keywords in the COM class name/description
   if ($defaultValue -and ($bitlockerKeywords | Where-Object { $defaultValue -like "*$_*" })) {
      # Check for LaunchPermission or security keys
      $launchPerm = Get-ItemProperty -Path $clsidKey -Name 'LaunchPermission' -ErrorAction SilentlyContinue
      $appId = (Get-ItemProperty -Path $clsidKey -Name 'AppID' -ErrorAction SilentlyContinue).'AppID'

      $bitlockerClsids += [PSCustomObject]@{
         CLSID   = $_.PSChildName
         Name    = $defaultValue
         AppID   = $appId
         HasLaunchPermission = ($null -ne $launchPerm)
         RegistryPath = $clsidKey
      }
   }
}

# Show all BitLocker-related COM CLSIDs (with possible Interactive User permissions)
$bitlockerClsids | Format-Table

# Optional, output to CSV for review
$bitlockerClsids | Export-Csv -Path "$env:USERPROFILE\Desktop\BitLocker_COM_CLSIDs.csv" -NoTypeInformation
```

This PowerShell Script will give you a list of all of the BitLocker related COM CLSIDs that have interactive user permissions, also at the end of the script there is an optional output to .csv file on the desktop.

3. Once you have run this, CHECK AND DOUBLE CHECK which are vulnerable or succeptable to the bitlockmove attack, note them for the next script.
4. Now paste the full {CLSID} into this script

```
# Written by N4Vx0.2 All Rights Reserved
# Requires administrator privileges
# Remove LaunchPermission from sensitive BitLocker COM CLSIDs (edit list as needed)
$bitlockerCLSIDs = @(
  "{D1ED435C-0A34-4591-BFFD-7CB276D78CE5}", # Fake CLSID please paste your CLSIDs that need permissions revoked here.
  "{F198B89A-5142-4204-ADF1-CB163E549798}", # Another Fake, please copy and paste the found dangerous CLSID's
  "{F198B89A-5142-4204-ADF1-CD163c549998}", # I have added extra spaces if you do not need them please remove
  "{F198B89A-5142-4204-ADF1-CD163c549998}",
  "{F198B89A-5142-4204-ADF1-CD163c549998}"
)

foreach ($clsid in $bitlockerCLSIDs) {
  $regPath = "HKLM:\SOFTWARE\Classes\CLSID\$clsid"
  try {
    Remove-ItemProperty -Path $regPath -Name "LaunchPermission" -ErrorAction SilentlyContinue
    Write-Host "LaunchPermission removed from $clsid"
  } catch {
    Write-Warning ("Could not remove LaunchPermission for " + $clsid + ": " + $_)
  }
}

# Recommend AppLocker setup if not configured (manual for most orgs)
# If AppLocker is not setup, please go look into it
$applockerPath = "HKLM:\SYSTEM\CurrentControlSet\Control\Srp\Gp\Exe"
if (-not (Test-Path $applockerPath)) {
  Write-Host "AppLocker is not configured. Please configure via Group Policy for organization-wide enforcement."
} else {
  Write-Host "AppLocker policies exist. Review them to ensure DLL rules block remote (UNC) paths."
}
```

Note: if you have not setup applocker on your device then you will get an error at the end of this script

5. Now we have just two more to go!
    a. Run this script to see what shares exist under SMB that have an “allow everyone” that are not system or Admin shares, and then attempts to revoke access for “everyone”

```
# Written by N4Vx0.2 All Rights Reserved
# User-Created SMB Share Hardening Script with CSV Export
# Only touches user-created shares, not system/admin shares

$adminSharePattern = '^[A-Z]\$$|^ADMIN\$|^IPC\$'  # Regex for system/admin shares
$shares = Get-SmbShare | Where-Object { $_.Name -notmatch $adminSharePattern }

$report = @()

foreach ($share in $shares) {
    $action = ""
    $errorMsg = ""
    try {
        # Check if "Everyone" has any access to the share
        $everyoneAccess = Get-SmbShareAccess -Name $share.Name | Where-Object { $_.AccountName -eq "Everyone" }
        if ($everyoneAccess) {
            # Remove "Everyone" access
            Revoke-SmbShareAccess -Name $share.Name -AccountName "Everyone" -Force -Confirm:$false
            $action = "Revoked access for 'Everyone'"
        } else {
            $action = "'Everyone' not present, no action needed"
        }
    } catch {
        $action = "Error"
        $errorMsg = $_.Exception.Message
        Write-Warning ("Error processing share {0}: {1}" -f $share.Name, $errorMsg)
    }

    $report += [PSCustomObject]@{
        ShareName     = $share.Name
        Path          = $share.Path
        Action        = $action
        ErrorMessage  = $errorMsg
    }
}

# Export to CSV on Desktop
$csvPath = [System.IO.Path]::Combine($env:USERPROFILE, 'Desktop', 'SMB_Share_Hardening_Report.csv')
$report | Export-Csv -Path $csvPath -NoTypeInformation

Write-Host "`nSummary report saved to: $csvPath"
```

Lastly, we go after bitpixie, if you see errors that means that PowerShell CANNOT make changes in your BIOS level system. Please note what it is saying to do that cannot be done and remediate it.

6. In the same PowerShell instance run the following script

```
# Written by N4vX0.2 All Rights Reserved
# Requires administrator privileges

# 1. Ensure Secure Boot is enabled
$sbstatus = Confirm-SecureBootUEFI
if ($sbstatus) {
   Write-Host "Secure Boot is ENABLED."
} else {
   Write-Warning "Secure Boot is NOT enabled! Please enable it in UEFI/BIOS setup."
}

# 2. Attempt to disable network boot (PXE) via registry (where supported)
# Note: Most network boot configs require BIOS/UEFI access, not just OS-side. This will only work on some vendors.
$regPath = "HKLM:\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters"
Set-ItemProperty -Path $regPath -Name "DisableDHCPMediaSense" -Value 1

# 3. Check and recommend kernel DMA protection (Windows 10/11+)
$dmaProt = Get-CimInstance -ClassName Win32_DeviceGuard | Select-Object -ExpandProperty SecurityServicesConfigured
if ($dmaProt -contains 2) {
   Write-Host "Kernel DMA Protection is ENABLED."
} else {
   Write-Warning "Kernel DMA Protection is NOT enabled! Please enable in firmware/OS."
}

# 4. Apply latest Secure Boot and BitLocker updates
Write-Host "Checking for and installing latest Windows updates (including BitLocker/Secure Boot)..."
Install-Module -Name PSWindowsUpdate -Force -ErrorAction SilentlyContinue
Import-Module PSWindowsUpdate
Get-WindowsUpdate -AcceptAll -Install -AutoReboot

Write-Host "Bitpixie mitigations complete. Please verify network boot options are DISABLED in UEFI/BIOS."
```

And Phew we are done!