Runbook: WinRM Setup and File Transfer

Version: 1.0 **Date:** 2025-11-11 **Purpose:** Configure and use Windows Remote Management (WinRM) for secure file transfers between jumper server and eKVM devices

---

Configuration Custody Notice
Ionic Health engineering exclusively manages all eKVM configuration, firmware, and software changes, including patching operations executed through the jumper server. LVHN operations is solely responsible for provisioning, hardening, and maintaining the Windows jumper server environment per the documented requirements.

Network Visibility Scope
Procedures here cover only the LVHN jumper server endpoint; Ionic Health does not require access to other LVHN network assets.

Overview

Windows Remote Management (WinRM) is Microsoft's implementation of the WS-Management protocol, allowing remote management of Windows systems. This runbook covers: - WinRM configuration on eKVM devices - TrustedHosts setup on jumper server - Secure file transfer using PowerShell Remoting - Troubleshooting common issues

---

Prerequisites

- ☐ Administrator access to both jumper server and eKVM
- ☐ Network connectivity on ports 5985 (HTTP) or 5986 (HTTPS)
- ☐ PowerShell 5.1 or later
- ☐ Valid domain or local credentials

Part 1: WinRM Configuration on eKVM

1.1 Enable WinRM Service

```
# Run on eKVM as Administrator
Enable-PSRemoting -Force

# Verify service is running
Get-Service WinRM | Select-Object Name, Status, StartType
# Expected: Status=Running, StartType=Automatic
```

1.2 Configure WinRM for HTTPS (Recommended)

```
# Check for existing HTTPS listener
Get-ChildItem WSMan:\localhost\Listener | Where-Object {$_.Keys -contains "Transport=HTTPS"}

# If no HTTPS listener, create self-signed certificate
$cert = New-SelfSignedCertificate -DnsName "ekvm-device-01.lvhn.local" `
                                  -CertStoreLocation "Cert:\LocalMachine\My" `
                                  -NotAfter (Get-Date).AddYears(5)

# Create HTTPS listener
New-Item -Path WSMan:\localhost\Listener `
         -Transport HTTPS `
         -Address * `
         -CertificateThumbprint $cert.Thumbprint -Force

# Verify listener
Get-ChildItem WSMan:\localhost\Listener | Select-Object Keys, Address, Enabled
```

1.3 Configure Firewall Rules

```
# Allow WinRM HTTP (port 5985)
New-NetFirewallRule -Name "WinRM-HTTP-In-TCP" `
                    -DisplayName "Windows Remote Management (HTTP-In)" `
                    -Enabled True `
                    -Direction Inbound `
                    -Protocol TCP `
                    -LocalPort 5985 `
                    -Action Allow `
                    -Profile Domain

# Allow WinRM HTTPS (port 5986) - Recommended
New-NetFirewallRule -Name "WinRM-HTTPS-In-TCP" `
                    -DisplayName "Windows Remote Management (HTTPS-In)" `
                    -Enabled True `
                    -Direction Inbound `
                    -Protocol TCP `
                    -LocalPort 5986 `
                    -Action Allow `
                    -Profile Domain

# Verify rules
Get-NetFirewallRule -Name "WinRM*" | Select-Object Name, Enabled, Direction, Action
```

1.4 Security Hardening

```
# Set authentication to Kerberos only (if domain-joined)
Set-Item WSMan:\localhost\Service\Auth\Kerberos -Value $true
Set-Item WSMan:\localhost\Service\Auth\Basic -Value $false

# Restrict to specific IP ranges (jumper server only)
$jumperIP = "10.50.100.10"
Set-Item WSMan:\localhost\Service\IPv4Filter -Value $jumperIP

# Enable credential delegation (if needed for double-hop scenarios)
Set-Item WSMan:\localhost\Client\Auth\CredSSP -Value $true

# Verify configuration
Get-ChildItem WSMan:\localhost\Service\Auth
Get-Item WSMan:\localhost\Service\IPv4Filter
```

Part 2: Jumper Server Configuration

2.1 Configure TrustedHosts

Note: Only required if eKVM is not domain-joined or using IP addresses instead of FQDNs.

```
# Run on jumper server as Administrator
# Option A: Add specific eKVM hostname
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "ekvm-device-01.lvhn.local" -Force

# Option B: Add multiple hosts (comma-separated)
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "ekvm-01,ekvm-02,ekvm-03" -Force

# Option C: Add subnet (use with caution)
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "10.60.200.*" -Force

# Verify TrustedHosts
Get-Item WSMan:\localhost\Client\TrustedHosts
```

2.2 Test WinRM Connectivity

```
# Test HTTP (port 5985)
Test-WSMan -ComputerName ekvm-device-01.lvhn.local

# Test HTTPS (port 5986)
Test-WSMan -ComputerName ekvm-device-01.lvhn.local -UseSSL

# Expected output: ProductVersion, ProtocolVersion, wsmid
```

```
# If error: "The client cannot connect to the destination specified in the request"
# → Check firewall, service status, TrustedHosts
```

Part 3: File Transfer Operations

3.1 Establish PowerShell Session

```
# Prompt for credentials
$ekvmCred = Get-Credential -Message "Enter eKVM credentials (LVHN\username or .\localadmin)"

# Create session (HTTP)
$session = New-PSSession -ComputerName ekvm-device-01.lvhn.local -Credential $ekvmCred

# Create session (HTTPS - recommended)
$sessionOptions = New-PSSessionOption -SkipCACheck -SkipCNCheck
$session = New-PSSession -ComputerName ekvm-device-01.lvhn.local `
                         -Credential $ekvmCred `
                         -UseSSL `
                         -SessionOption $sessionOptions

# Verify session state
$session | Select-Object Id, ComputerName, State, Availability
# Expected: State=Opened, Availability=Available
```

3.2 Transfer File (Jumper → eKVM)

```
# Copy single file
Copy-Item -Path "C:\Staging\eKVM\firmware.exe" `
          -Destination "C:\Temp\" `
          -ToSession $session `
          -Verbose

# Copy multiple files
Copy-Item -Path "C:\Staging\eKVM\*" `
          -Destination "C:\Temp\" `
          -ToSession $session `
          -Recurse `
          -Verbose

# Verify file arrived
Invoke-Command -Session $session -ScriptBlock {
    Get-ChildItem -Path "C:\Temp\firmware.exe" | Select-Object Name, Length, LastWriteTime
}
```

3.3 Transfer File (eKVM → Jumper)

```
# Copy from remote to local
Copy-Item -Path "C:\Logs\install.log" `
          -Destination "C:\Evidence\" `
          -FromSession $session `
          -Verbose
```

3.4 Run Commands on eKVM

```
# Execute command and get output
$result = Invoke-Command -Session $session -ScriptBlock {
    Get-Service -Name "eKVM*" | Select-Object Name, Status
}
$result | Format-Table -AutoSize

# Execute script file on remote system
Invoke-Command -Session $session -FilePath "C:\Scripts\Post-Install-Validation.ps1"
```

3.5 Close Session

```
# Remove session (cleanup)
Remove-PSSession -Session $session

# Or remove all sessions
Get-PSSession | Remove-PSSession
```

Part 4: Advanced Scenarios

4.1 Large File Transfer with Progress

```
# For files >100 MB, use compression
$sourceFile = "C:\Staging\eKVM\large-firmware.exe"
$destPath = "C:\Temp\"

# Get file size
$fileSize = (Get-Item $sourceFile).Length
Write-Host "Transferring $($fileSize / 1MB) MB..." -ForegroundColor Cyan

# Transfer with timeout
Copy-Item -Path $sourceFile `
          -Destination $destPath `
          -ToSession $session `
          -Verbose `
          -ErrorAction Stop
```

```
Write-Host "Transfer complete!" -ForegroundColor Green
```

4.2 Multi-Device Transfer (Parallel)

```
$ekvmDevices = @("ekvm-01", "ekvm-02", "ekvm-03")
$sourceFile = "C:\Staging\eKVM\firmware.exe"

$ekvmDevices | ForEach-Object -Parallel {
    $device = $_
    $cred = $using:ekvmCred
    $file = $using:sourceFile

    try {
        $session = New-PSSession -ComputerName $device -Credential $cred
        Copy-Item -Path $file -Destination "C:\Temp\" -ToSession $session -Verbose
        Remove-PSSession -Session $session
        Write-Host "[SUCCESS] $device" -ForegroundColor Green
    } catch {
        Write-Host "[FAILED] $device : $_" -ForegroundColor Red
    }
} -ThrottleLimit 5
```

4.3 Integrity Verification During Transfer

```
$sourceFile = "C:\Staging\eKVM\firmware.exe"
$expectedHash = "a1b2c3d4e5f67890123456789012345678901234567890abcdef1234567890abcdef123456"

# Hash before transfer
$hashBefore = (Get-FileHash -Path $sourceFile -Algorithm SHA256).Hash
if ($hashBefore -ne $expectedHash) {
    Write-Error "Source file hash mismatch!"
    exit 1
}

# Transfer
Copy-Item -Path $sourceFile -Destination "C:\Temp\" -ToSession $session -Verbose

# Hash after transfer
$hashAfter = Invoke-Command -Session $session -ScriptBlock {
    (Get-FileHash -Path "C:\Temp\firmware.exe" -Algorithm SHA256).Hash
}

# Compare
if ($hashAfter -eq $expectedHash) {
    Write-Host "[PASS] Integrity verified" -ForegroundColor Green
```

```
} else {
    Write-Host "[FAIL] Hash mismatch!" -ForegroundColor Red
}
```

Part 5: Troubleshooting

5.1 Connection Refused / Timeout

Symptoms:

```
New-PSSession : [ekvm-device-01] Connecting to remote server ekvm-device-01 failed with the
The client cannot connect to the destination specified in the request.
```

Resolution:

```
# 1. Check service running
Get-Service WinRM -ComputerName ekvm-device-01  # Requires RPC/135 open

# 2. Test port connectivity
Test-NetConnection -ComputerName ekvm-device-01 -Port 5985

# 3. Check firewall rule
Get-NetFirewallRule -Name "WinRM-HTTP-In-TCP" | Select-Object Enabled, Direction, Action

# 4. Enable WinRM (if disabled)
Enable-PSRemoting -Force
```

5.2 Access Denied

Symptoms:

```
New-PSSession : [ekvm-device-01] Connecting to remote server ekvm-device-01 failed with the
Access is denied.
```

Resolution:

```
# 1. Verify credentials
whoami  # On local machine
Invoke-Command -ComputerName ekvm-01 -Credential $cred -ScriptBlock {whoami}  # On remote

# 2. Check local admin membership
Invoke-Command -ComputerName ekvm-01 -Credential $cred -ScriptBlock {
    Get-LocalGroupMember -Group "Administrators"
}

# 3. Check WinRM authentication
Get-ChildItem WSMan:\localhost\Service\Auth
```

```
# 4. Try CredSSP (if Kerberos fails)
Set-Item WSMan:\localhost\Client\Auth\CredSSP -Value $true
$session = New-PSSession -ComputerName ekvm-01 -Credential $cred -Authentication CredSSP
```

5.3 TrustedHosts Error

Symptoms:

```
New-PSSession : [ekvm-device-01] Connecting to remote server ekvm-device-01 failed with the
The WinRM client cannot process the request. If the authentication scheme is different from
```

Resolution:

```
# Add to TrustedHosts
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "ekvm-device-01" -Force -Concatenate

# Verify
Get-Item WSMan:\localhost\Client\TrustedHosts
```

5.4 File Transfer Hangs

Symptoms: - `Copy-Item` runs indefinitely without completing - No error messages

Resolution:

```
# 1. Check session state
$session | Select-Object State, Availability
# If State=Disconnected, recreate session

# 2. Test with small file first
$testFile = New-Object byte[] 1MB
[System.IO.File]::WriteAllBytes("C:\test-1MB.bin", $testFile)
Copy-Item -Path "C:\test-1MB.bin" -Destination "C:\Temp\" -ToSession $session

# 3. Increase MaxMemoryPerShellMB (if large file)
Invoke-Command -Session $session -ScriptBlock {
    Set-Item WSMan:\localhost\Shell\MaxMemoryPerShellMB -Value 2048
    Restart-Service WinRM
}

# 4. Use SMB as fallback (see SMB runbook)
```

5.5 Certificate Errors (HTTPS)

Symptoms:

```
Test-WSMan : <f:WSManFault><f:Message>The certificate presented by the remote computer ekvm-
```

Resolution:

```powershell
# Option A: Skip certificate validation (less secure, testing only)
$sessionOptions = New-PSSessionOption -SkipCACheck -SkipCNCheck -SkipRevocationCheck
$session = New-PSSession -ComputerName ekvm-01 -Credential $cred -UseSSL -SessionOption $ses

# Option B: Install certificate (production)
# 1. Export certificate from eKVM
$cert = Get-ChildItem Cert:\LocalMachine\My | Where-Object {$_.Subject -match "ekvm-device-0
Export-Certificate -Cert $cert -FilePath "C:\ekvm-cert.cer"

# 2. Import on jumper server
Import-Certificate -FilePath "C:\ekvm-cert.cer" -CertStoreLocation Cert:\LocalMachine\Root
```

Part 6: Security Best Practices

6.1 Restrict TrustedHosts

```powershell
# BAD: Allow all hosts
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "*"  #  Never do this

# GOOD: Specific hostnames
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "ekvm-01,ekvm-02,ekvm-03"  #

# BETTER: Use domain authentication (no TrustedHosts needed)
# Ensure both systems are domain-joined
```

6.2 Use HTTPS (Port 5986)

```powershell
# Always prefer HTTPS over HTTP
$session = New-PSSession -ComputerName ekvm-01 -Credential $cred -UseSSL
```

6.3 Limit Session Duration

```powershell
# Set idle timeout (15 minutes)
Set-Item WSMan:\localhost\Shell\IdleTimeout -Value 900000

# Set max session lifetime (2 hours)
Set-Item WSMan:\localhost\Shell\MaxShellRunTime -Value 7200000
```

6.4 Enable Logging

```powershell
# Enable PowerShell script block logging
Set-ItemProperty -Path "HKLM:\SOFTWARE\Wow6432Node\Policies\Microsoft\Windows\PowerShell\Scr
                 -Name "EnableScriptBlockLogging" -Value 1

# Enable module logging
```

```
Set-ItemProperty -Path "HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell\ModuleLogging"
                 -Name "EnableModuleLogging" -Value 1
```

Part 7: Cleanup and Maintenance

7.1 Remove TrustedHosts Entries

```
# View current TrustedHosts
$current = (Get-Item WSMan:\localhost\Client\TrustedHosts).Value

# Remove specific entry
$updated = $current -replace "ekvm-device-01,", ""
Set-Item WSMan:\localhost\Client\TrustedHosts -Value $updated -Force

# Clear all TrustedHosts
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "" -Force
```

7.2 Disable WinRM (Post-Window)

```
# Stop service
Stop-Service WinRM

# Disable service
Set-Service WinRM -StartupType Disabled

# Or disable PS Remoting
Disable-PSRemoting -Force
```

7.3 Review WinRM Logs

```
# View WinRM operational log
Get-WinEvent -LogName "Microsoft-Windows-WinRM/Operational" -MaxEvents 50 | Format-Table -Au

# Export for audit
wevtutil epl "Microsoft-Windows-WinRM/Operational" "C:\Evidence\WinRM.evtx"
```

Part 8: Quick Reference

Essential Commands

```
# Enable WinRM
Enable-PSRemoting -Force

# Test connectivity
```

```
Test-WSMan -ComputerName <host>

# Create session
$s = New-PSSession -ComputerName <host> -Credential (Get-Credential)

# Copy file TO remote
Copy-Item -Path <source> -Destination <dest> -ToSession $s

# Copy file FROM remote
Copy-Item -Path <source> -Destination <dest> -FromSession $s

# Run command on remote
Invoke-Command -Session $s -ScriptBlock {<commands>}

# Close session
Remove-PSSession $s
```

Port Reference

Port	Protocol	Description
5985	HTTP	WinRM default (unencrypted)
5986	HTTPS	WinRM over SSL/TLS (recommended)

Event Log Reference

Event ID	Log	Description
6	Microsoft-Windows-WinRM/Operational	WSMan session created
8	Microsoft-Windows-WinRM/Operational	WSMan session closed
142	Microsoft-Windows-WinRM/Operational	WS-Management service started
161	Microsoft-Windows-WinRM/Operational	WinRM created listener

---

Related Documents

- MOP: eKVM Remote Update

- Runbook: SMB File Transfer
- Runbook: RDP Hardening
- Microsoft WinRM Documentation

Document Owner: Ionic Engineering Team **Last Updated:** 2025-11-11 **Review Cycle:** Quarterly