EXTRAHOP®

WHITE PAPER

Decrypting the Shadows: Adversaries Hiding Lateral Movements in the Modern Enterprise

TABLE OF CONTENTS

Introduction

The modern cyber defense landscape is defined by a central paradox: The very tools designed to facilitate seamless enterprise administration are also the most potent weapons in the arsenal of adversaries. In the modern enterprise environment, it's not a matter of if the network will be compromised, but a matter of when. Network security has shifted from the "front door" to the "invisible corridors" of internal movement.[1] Statistics from 2025 reveal the reality where "Living Off The Land Binaries, Scripts and Libraries" (LOLBAS)[2] techniques were implicated in 84% of attacks.[3] This approach allows adversaries to achieve their objectives without leaving the traditional forensic footprint, effectively blending into the "trusted noise" of everyday SOC operations.[4]

Among the most misused LOLBAS are Microsoft PowerShell and Windows Management Instrumentation (WMI)[5] framework over Remote Procedure Call (RPC)[6] or Web Services Management (WinRM/WSMAN)[7] protocols. These tools and services represent the central nervous system of a Windows-based enterprise, handling everything from configuration and monitoring to remote software deployment and Active Directory (AD) management. Because they are indispensable for core administrative functions such as managing Group Policy Objects (GPO), they cannot be disabled globally without catastrophic operational consequences. Furthermore, their inherent encryption, designed to protect sensitive administrative traffic, ironically provides a perfect cover for lateral movement. Understanding the granular mechanics of these services is essential for defense.

Effective enterprise network defense requires layered visibility, often achieved through deployment of the "Visibility Triad": endpoint detection and response (EDR), intrusion detection and prevention systems (IDPS), and network detection and response (NDR) appliances. While IDPS solutions offer perimeter-focused, signature-based matching for suspicious network patterns, they struggle with "East-West" lateral traffic, internal encrypted data, and LOLBAS tools used for malicious purposes. EDR provides deep insight into events on a compromised host, but lacks visibility into the broader network environment. For tracking lateral movement, an advanced NDR with decryption capabilities is critical and is considered the source of truth. NDR appliances continuously monitor network conversations, flagging malicious events by analyzing behavior and metadata. A key advantage of NDRs is that their resiliency and visibility are provided without software running on the monitored machines, making it difficult for attackers to circumvent or disable.

This white paper aims to demonstrate the necessity for decryption to detect lateral movement leveraging WMI over WinRM/WSMAN and RPC. We'll analyze both the original encrypted traffic and the post-decryption insights to expose lateral movement. Additionally, we'll examine how threat actors exploit these legitimate communication channels to hide their activity.

1. https://blogs.vmware.com/security/2026/02/zero-trust-journey-vdefend.html
2. https://lolbas-project.github.io/
3. https://www.bitdefender.com/content/dam/bitdefender/business/campaign/assessment/Official-2025-Cybersecurity-Assessment-Report.pdf
4. https://www.cisa.gov/sites/default/files/2025-03/Joint-Guidance-Identifying-and-Mitigating-LOTL508.pdf
5. https://learn.microsoft.com/en-us/windows/win32/wmisdk/wmi-start-page
6. https://www.extrahop.com/resources/protocols/wmi
7. https://www.extrahop.com/resources/protocols/wsman

The Components of Lateral Movement

The technical foundation of any modern Windows environment rests upon a stack of proprietary and industry-standard protocols that facilitate communication between systems. To understand why an adversary chooses these paths, it's important to understand their role as legitimate, essential services.

WMI (Windows Management Instrumentation) Framework

WMI is a database and service for managing data and operations on Windows-based operating systems. It acts as a mediator between management clients and system components.

- **The Repository & Providers:** WMI consists of a repository (storing class definitions) and providers (the code that performs the work).
- **Weaponization:** The most common technique involves the **Win32_Process** class. By calling the **Create** method of this class,[8] an attacker can spawn an arbitrary process (like **cmd.exe** or **powershell.exe**) on a remote system.

PowerShell: The Execution Engine

PowerShell is a powerful task-automation framework and a core interface for lateral movement. While widely used by professionals for system management and security automation, it is also leveraged by attackers to gain unauthorized access and execute malicious code. PowerShell frequently interacts directly with WSMAN and other protocols, including WMI over WSMAN, and various Windows API's, which allows attackers to run scripts without leaving a trace.

- **Adversary Use:** Attackers use PowerShell (**T1059.001**) to execute obfuscated scripts, load malware into memory via reflection (bypassing disk-based detection), and orchestrate remote commands[9] using built-in cmdlets like **Invoke-Command**, **Invoke-WmiMethod,**[10] **and Invoke-Expression**.

WMI over MS-RPC

Microsoft Remote Procedure Call (MS-RPC) is the underlying technology used by WMI via the Distributed Component Object Model (DCOM).

- **How it Works:** RPC functions by connecting to specific interfaces and then utilizing operation numbers (opnums) to execute designated functions on a target server.
- **Port Dependency:** WMI over RPC movement requires port 135 (RPC Endpoint Mapper) and a dynamic range of high ports (49152–65535) for DCOM communication.
- **Adversary Use:** Attackers exploit the underlying DCE/RPC runtime to execute sensitive administrative functions, often bypassing high-level security monitoring like EDRs (**T1047**).

WSMAN and WinRM: The Transport

WS-Management (WSMAN) is an industry-standard, SOAP-based protocol for managing systems via HTTP/HTTPS. WinRM is Microsoft's implementation of this protocol.

8. https://www.cybereason.com/blog/wmi-lateral-movement-win32
9. https://redcanary.com/threat-detection-report/techniques/powershell/
10. https://huntsman-dfir.medium.com/lateral-movement-with-wmi-windows-management-instrumentation-67639e9456cb

- **How it Works:** WMI commands are passed over WSMAN through built-in functionality within Windows such as WinRS or WinRM. The communication is encrypted with Kerberos/NTLM (port 5985) or TLS (port 5986).
- **Adversary Use:** WinRM is frequently enabled Windows enterprise environments to aid in remote administration. Attackers with valid credentials can establish a persistent, encrypted remote shell that looks exactly like legitimate administrative traffic (**T1021.006**). Because raw RPC is often heavily firewalled or monitored, adversaries may prefer moving laterally via WinRM/WSMAN because its ports are more likely to be open for cross-segment management.

Deep Dive: PowerShell Using WMI over MS-RPC/DCOM

Unlike modern web-based protocols, WMI can utilize the MS-RPC/DCOM framework, which introduces significant complexity for network defenders due to its dynamic nature and creates the perfect hiding place for attacker lateral movement efforts. A WMI connection typically begins on TCP port 135, with a request to the RPC Endpoint Mapper (EPM). The EPM serves as a directory service, telling the client which dynamic, high-numbered TCP port has been assigned to the requested WMI service. Following this, the client establishes a new connection to the target system on the dynamic high port.

The WMI service interacts with WMI providers (e.g., **Win32_Process** or **Win32_Service**) to access or modify managed components (running process, registry key, installed service, file information, etc.).

The most common technique for Remote Code Execution (RCE) via WMI is the abuse of the **Win32_Process** class. In this technique, an attacker remotely invokes **Win32_Process::Create** to create a new process based on the command specified in the **CommandLine** parameter.

```
PS C:\Users\domainadmin> Invoke-WmiMethod -Class Win32_Process -Name Create -ArgumentList $FinalCommand -ComputerName srv-dc-0 -Credential (Get-Credential)

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential


__GENUS          : 2
__CLASS          : __PARAMETERS
__SUPERCLASS     :
__DYNASTY        : __PARAMETERS
__RELPATH        :
__PROPERTY_COUNT : 2
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
ProcessId        : 4628
ReturnValue      : 0
PSComputerName   :


PS C:\Users\domainadmin>
```

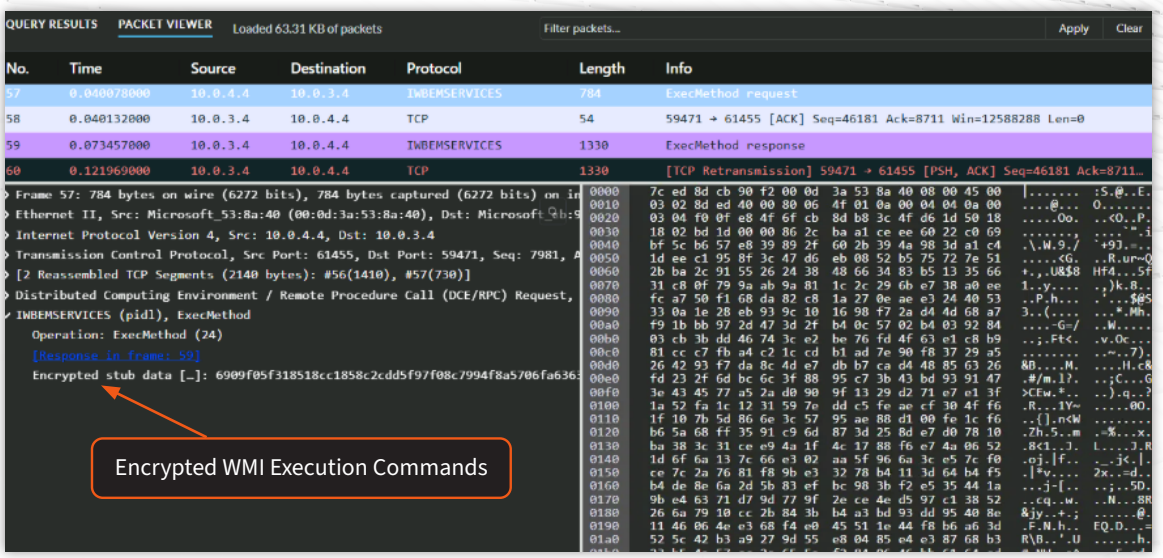
Example of an attacker remotely invoking Win32_Process::Create

Fileless Payload Delivery Technique

By manipulating a service's **PathName**, attackers employ a fileless payload delivery technique.[11] This property is configured to execute an encoded PowerShell command (e.g., **cmd.exe /c powershell.exe -EncodedCommand**). Following the service's creation, the attacker invokes the **StartService** method to execute the payload. Once it is completed, the **Delete** method is called on the service object for anti-forensic cleanup.

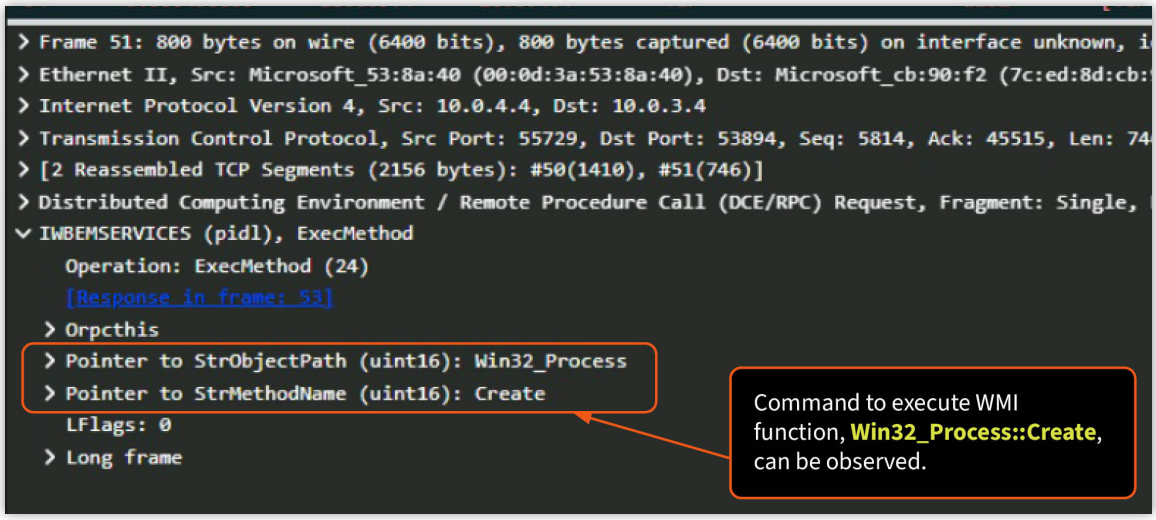
For a security analyst, monitoring the relationship between the initial port (135) request and the subsequent high-numbered port connection is a key behavioral indicator of WMI activity. However, without a decryption capability, inspection effectively stops here. The specific method being called (e.g., create vs. a benign information query) remains unknown.

11. https://blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf



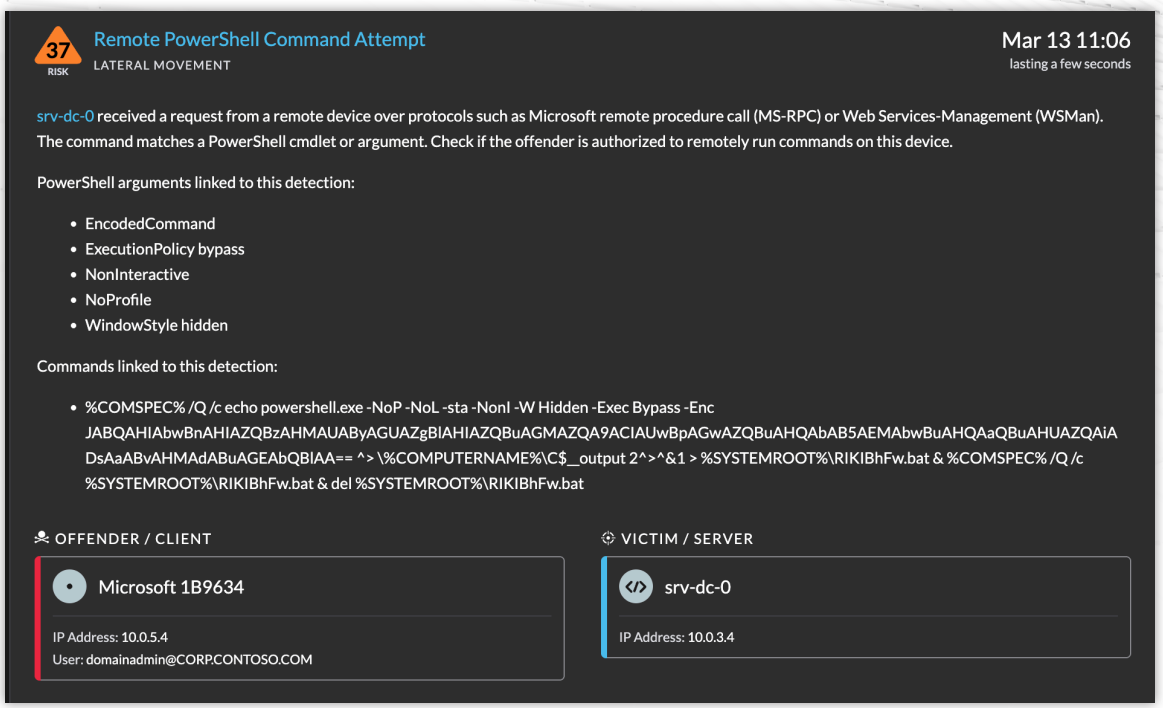
Without decrypting capabilities DCOM/RPC traffic with WMI execution commands is encrypted.

An NDR solution, such as ExtraHop's RevealX™ 360, unravels encrypted payloads by decrypting RPC traffic, thereby exposing structured protocol information. This is where detecting the usage of PowerShell in these encrypted communications becomes a first-class detection mechanism.



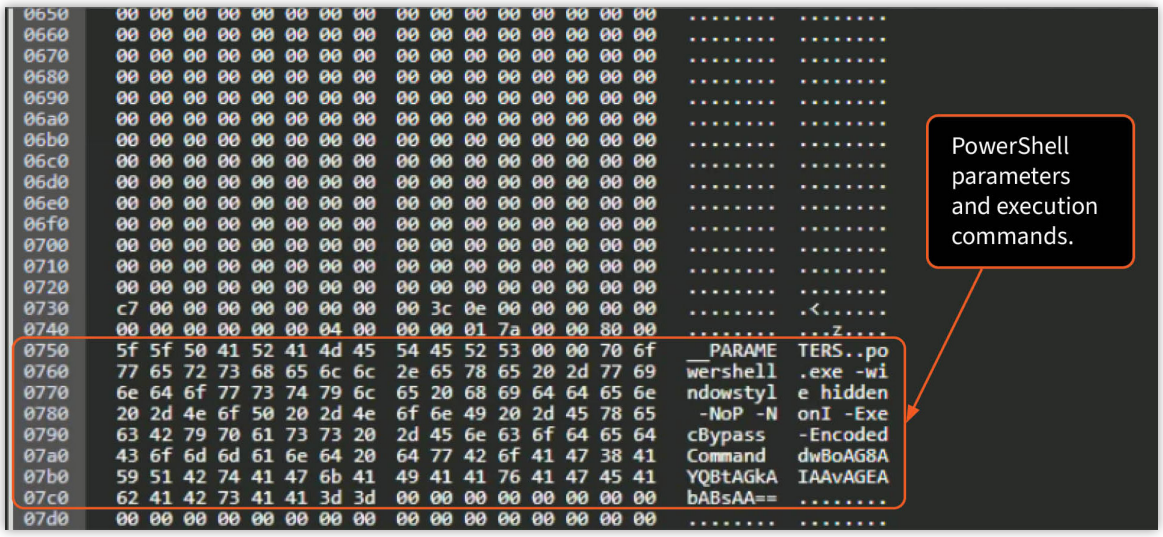
With decryption capabilities, DCOM/RPC trffic reveals actions, such as create a WMI function command.

Traditional NDR solutions are often inadequate for analyzing RPC traffic, typically relying only on the associated **opnum** and interface to trigger an event, thus keeping the true intent of the activity hidden. Adversaries frequently employ PowerShell with obfuscation flags (e.g., -**NoP, -W Hidden**, or **-Enc**) to evade detection. RevealX overcomes this limitation by enabling decryption, which exposes the command-line arguments concealed within the encrypted RPC traffic, leading to high-fidelity alerts.



RevealX capability to decrypt RPC or WSMan traffic containing PowerShell commands containing arguments such as -NoP.

At the final layer, the payload itself is visible. Decrypted traffic exposes payloads in plaintext, including full commands and arguments the actor executed. Encoded PowerShell, in-memory execution logic, and arguments passed to native tools are visible before execution begins. For a network defender, this means the actor's execution intent is no longer inferred, it is observed.



With decryption capabilities, PowerShell parameters and commands are visible.

Deep Dive: PowerShell over WSMAN-Based WinRM

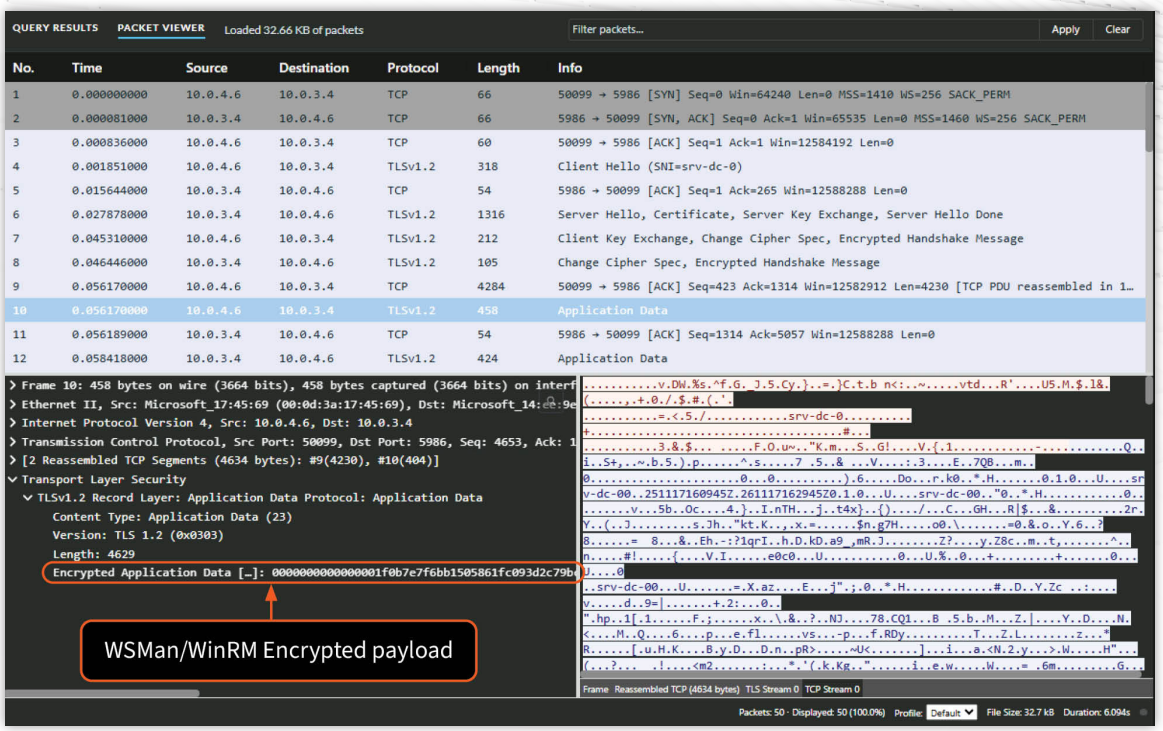
WinRM is Microsoft's proprietary implementation of the open source standard WSMAN, which is used to manage and interact with remote machines. WinRM/WSMan utilize Simple Object Access Protocol (SOAP) to exchange data over web services like HTTP/HTTPS using eXtensible Markup Language (XML) for formatting. The XML is where the actual payload, such as the **Get-Service** command, is serialized into XML, encrypted using a session key negotiated via Kerberos, NTLM, or TLS[12] and finally transmitted over HTTP (port 5985) or HTTPS (port 5986).

There are two methods that allow attackers to execute commands remotely. WMI can be carried over WSMAN and PowerShell Remoting Protocol (PSRP). The primary difference of use is based on attacker sophistication and scalability:

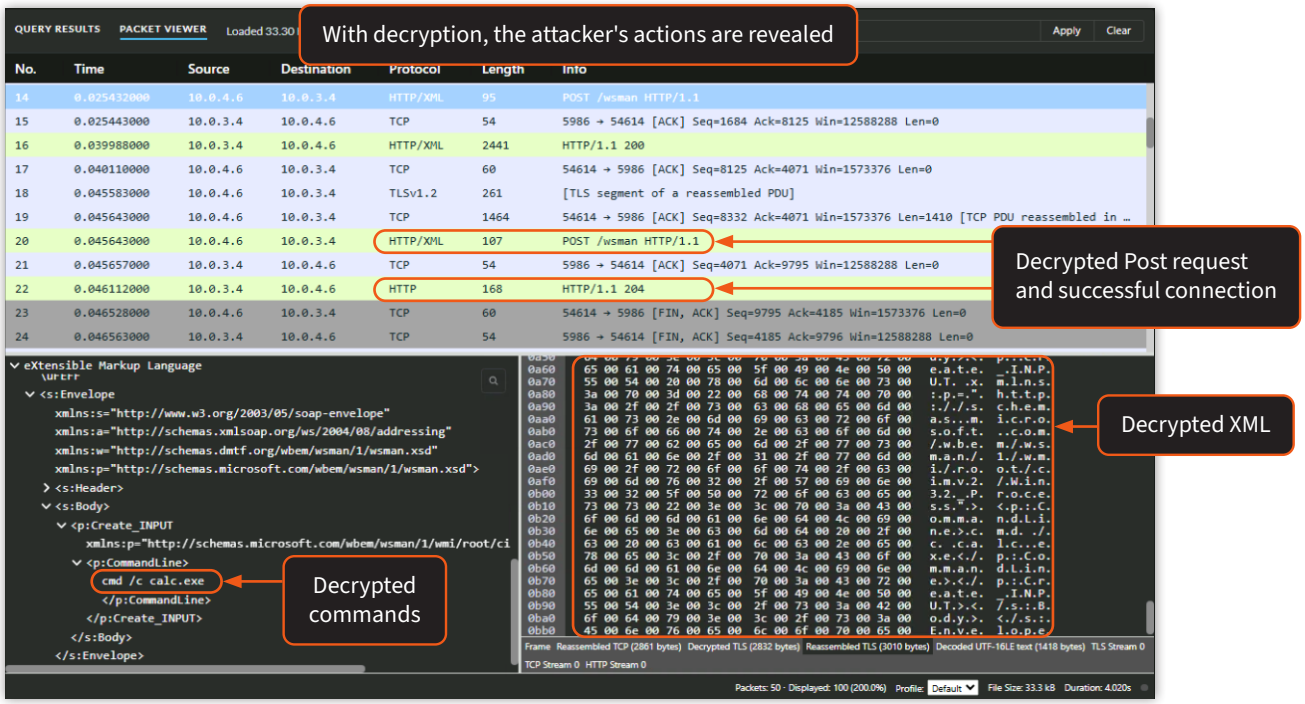
- **WinRM** enables remote management of a server and is Microsoft's implementation of WSMAN.
- **WinRS** is the native client-side application to facilitate communication over WSMAN. WinRS offers some command-line monitoring evasion as WinRS executes commands via '**cmd.exe**' or direct binary execution, which often bypasses PowerShell-centric detections. Additionally, sessions are short-lived, terminating immediately following command execution.
- **PSRP** allows an attacker to run complex scripts in memory with full access to the PowerShell suite. Attackers initiate a PSRP session using the **Enter-PSSession** cmdlet. This process first uses WinRM/WSMAN[13] and then leverages WMI commands through cmdlets (e.g., **Invoke-Command**). This allows a payload to be quickly distributed across multiple hosts. Unlike WinRS, PSRP supports persistent sessions, allowing attackers to navigate to and from target environments without reauthenticating and creating new logon events.

Without the ability to decrypt this traffic, network-based security tools are limited to observing standard HTTP traffic with high-entropy payloads. This provides an evasion mechanism for attackers. Advanced NDR solutions, such as RevealX, can integrate with AD for decrypting Kerberos or NTLM-encrypted traffic. With session key forwarding, RevealX is also capable of peering into TLS-encrypted traffic. Decryption is crucial for revealing specific adversarial actions, such as the underlying PowerShell commands being sent (e.g., **Invoke-Expression** or **Invoke-Command -ScriptBlock {... }**).

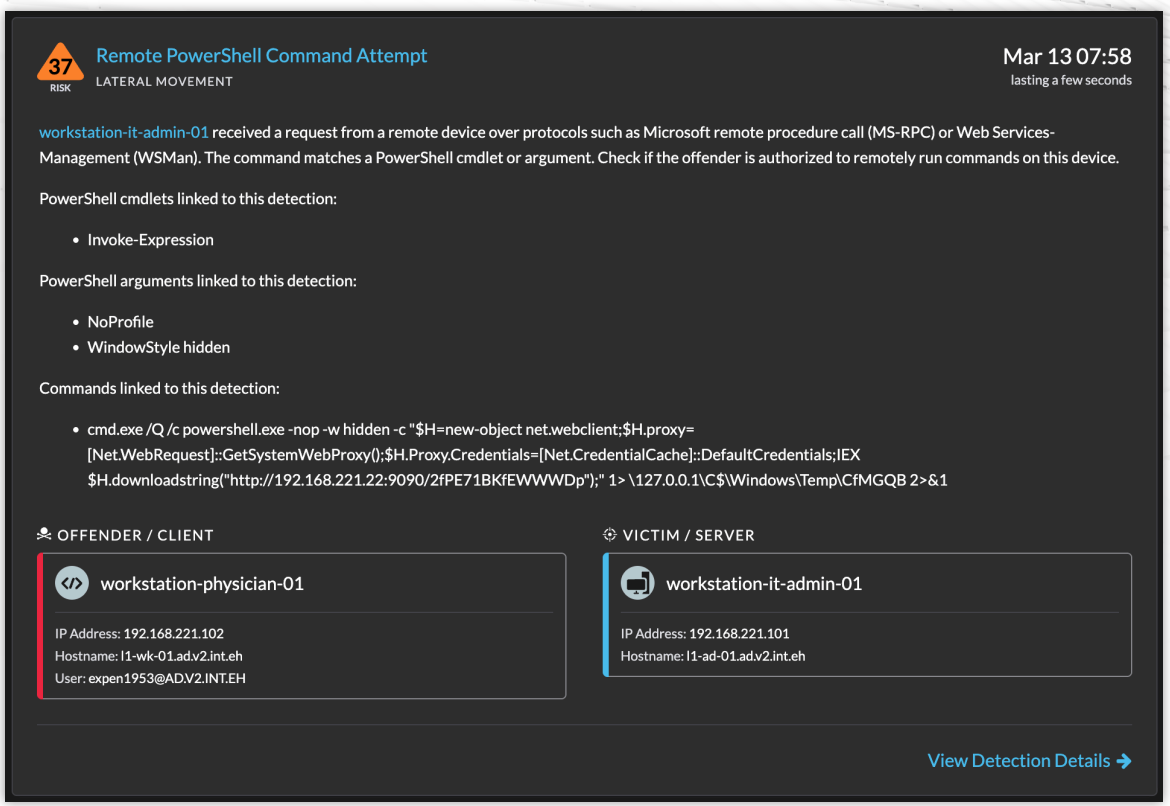
12. https://learn.microsoft.com/en-us/powershell/scripting/security/remoting/winrm-security?view=powershell-7.5
13. https://learn.microsoft.com/en-us/powershell/scripting/security/remoting/winrm-security?view=powershell-7.5



Encrypted WSMan via TLS traffic hides in an attacker's commands.



RevealX can decrypt TLS traffic exposing plaintext XML data. This example shows an attacker launching the calculator application.



RevealX detection event triggering on decrypted MS-RPC or WSMan traffic using PowerShell.

Decrypting PSRP Traffic

As previously mentioned, PRSP utilizes a persistent session between a client and server. It operates over WSMAN. An example of a decrypted PSRP HTTP header is below:

- **Method: HTTP POST**
- **URI: /wsman?PSVersion=**x.x.xxxxx.xxxx
- **Content-Type: application/soap+xml;charset=**UTF-16
- **User-Agent:** Microsoft WinRM Client
- **Host: HOSTNAME**:{**5985|5986**}

Attackers frequently use various tools, such as Evil-WinRM or OpenWSMAN, to communicate over WSMAN. Each of these tools has a unique, though easily modifiable, User-Agent string defined in its source code. A key differentiator is that for these tools the HTTP Content-Type '**charset=**' parameter has been observed to be UTF-8. Using WinRS, the URI is '**/wsman**'. Other HTTP Header characteristics remain consistent across these methods.

A common mistake is assuming one POST equals one command. In reality, a single LOLBAS (**T1543.003**) execution triggers a lifecycle visible in the XML **<a:Action>** headers. In decrypted PSRP traffic, the embedded XML payload reveals three URI variations that map an attacker's progress:

1. **Command:**…/**windows/shell/command**
   The "Initiation." This request tells the remote host to open a shell or execute a process (e.g., calling **powershell.exe** or **cmd.exe**).
2. **Signal:** …**windows/shell/signal**
   The "Polling." Because WinRM is request/response, the client must repeatedly "ask" the server for the output of the command it just started.
3. **Receive:**…**/windows/shell/receive**
   The "Teardown." This terminates the process or communicates state changes (like a Ctrl+C interrupt).

The "On-the-Wire" Exchange Flow

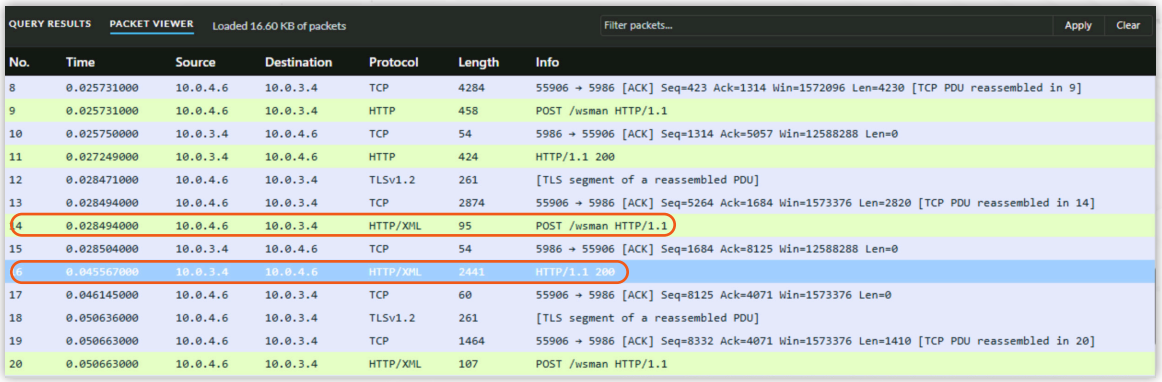
When an actor executes a LOLBAS command, the network exchange follows a predictable, structured pattern. Monitoring the transition from authentication to execution is where detections are won.

Table 1: WinRM traffic flow from authentication to execution

Phase	Direction	Protocol	Insight for defenders
Authentication	Sender →	HTTP POST (KRB5 AP-REQ)	The initial Kerberos ticket request to establish identity.
Authentication	Recipient ←	HTTP/1.1 200 (KRB5 AP-REP)	The session is now authenticated through a successful connection.
Execution	Sender →	HTTP POST (LOLBAS Payload)	The encrypted XML blob contains the malicious command.
Status	Recipient ←	HTTP/1.1 200 (ProcessID)	The server returns the PID and ReturnValue.
Cleanup	Sender →	HTTP POST (Empty Body)	Final synchronization using random UUIDs in the XML.
Closure	Recipient ←	HTTP/1.1 204 (No Content)	The individual execution is complete.

Decrypting WSMAN: Extracting the “Smoking Gun”

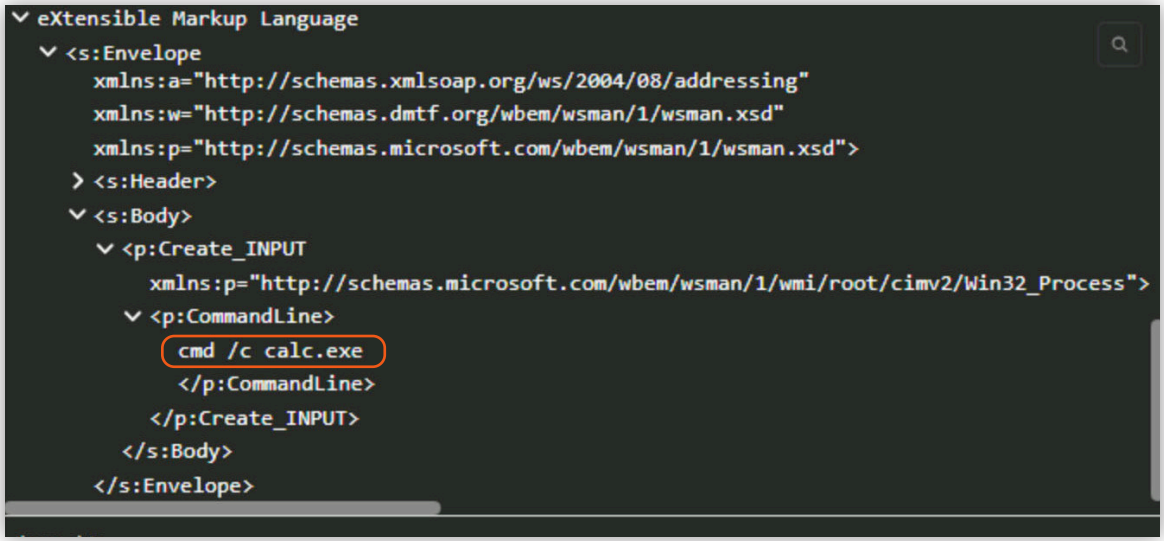
Full decryption of WSMAN is essential because it reveals the ScriptBlock content. NDR platforms that integrate with AD can retrieve the necessary Kerberos/NTLM or TLS keys to decrypt this traffic.[14] This allows the defender to see the attacker’s actions as depicted in images 9 through 11. In this example, the attack leverages WSMAN to launch the built-in calculator application and retrieve the associated process ID for further exploitation.



QUERY RESULTS | PACKET VIEWER | Loaded 16.60 KB of packets | Filter packets... | Apply | Clear

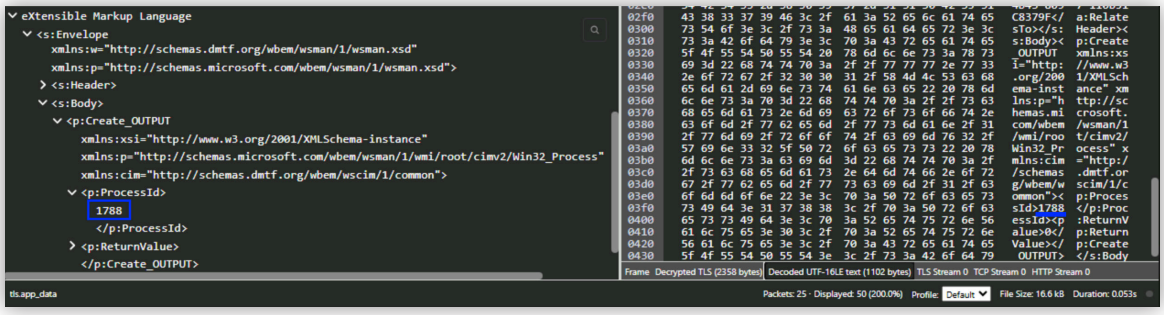
No.	Time	Source	Destination	Protocol	Length	Info
8	0.025731000	10.0.4.6	10.0.3.4	TCP	4284	55906 → 5986 [ACK] Seq=423 Ack=1314 Win=1572096 Len=4230 [TCP PDU reassembled in 9]
9	0.025731000	10.0.4.6	10.0.3.4	HTTP	458	POST /wsman HTTP/1.1
10	0.025750000	10.0.3.4	10.0.4.6	TCP	54	5986 → 55906 [ACK] Seq=1314 Ack=5057 Win=12588288 Len=0
11	0.027249000	10.0.3.4	10.0.4.6	HTTP	424	HTTP/1.1 200
12	0.028471000	10.0.4.6	10.0.3.4	TLSv1.2	261	[TLS segment of a reassembled PDU]
13	0.028494000	10.0.4.6	10.0.3.4	TCP	2874	55906 → 5986 [ACK] Seq=5264 Ack=1684 Win=1573376 Len=2820 [TCP PDU reassembled in 14]
14	0.028494000	10.0.4.6	10.0.3.4	HTTP/XML	95	POST /wsman HTTP/1.1
15	0.028504000	10.0.3.4	10.0.4.6	TCP	54	5986 → 55906 [ACK] Seq=1684 Ack=8125 Win=12588288 Len=0
16	0.045567000	10.0.3.4	10.0.4.6	HTTP/XML	2441	HTTP/1.1 200
17	0.046145000	10.0.4.6	10.0.3.4	TCP	60	55906 → 5986 [ACK] Seq=8125 Ack=4071 Win=1573376 Len=0
18	0.050636000	10.0.4.6	10.0.3.4	TLSv1.2	261	[TLS segment of a reassembled PDU]
19	0.050663000	10.0.4.6	10.0.3.4	TCP	1464	55906 → 5986 [ACK] Seq=8332 Ack=4071 Win=1573376 Len=1410 [TCP PDU reassembled in 20]
20	0.050663000	10.0.4.6	10.0.3.4	HTTP/XML	107	POST /wsman HTTP/1.1

Decrypted WSMan traffic showing the attacker’s POST /wsman.



```
eXtensible Markup Language
  <s:Envelope
     xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
     xmlns:w="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
     xmlns:p="http://schemas.microsoft.com/wbem/wsman/1/wsman.xsd">
    <s:Header>
    <s:Body>
      <p:Create_INPUT
         xmlns:p="http://schemas.microsoft.com/wbem/wsman/1/wmi/root/cimv2/Win32_Process">
        <p:CommandLine>
          cmd /c calc.exe
          </p:CommandLine>
        </p:Create_INPUT>
      </s:Body>
    </s:Envelope>
```

Packet Details pane showing decrypted command to run ‘calc.exe’.



```
eXtensible Markup Language
  <s:Envelope
     xmlns:w="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
     xmlns:p="http://schemas.microsoft.com/wbem/wsman/1/wsman.xsd">
    <s:Header>
    <s:Body>
      <p:Create_OUTPUT
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:p="http://schemas.microsoft.com/wbem/wsman/1/wmi/root/cimv2/Win32_Process"
         xmlns:cim="http://schemas.dmtf.org/wbem/wscim/1/common">
        <p:ProcessId>
          1788
          </p:ProcessId>
        <p:ReturnValue>
        </p:Create_OUTPUT>
```

```
02f0  43 38 33 37 39 46 3c 2f  61 3a 52 65 6c 61 74 65   C8379F</ a:Relate
0300  73 54 6f 3e 3c 2f 73 3a  48 65 61 64 65 72 3e 3c   sTo></s: Header><
0310  73 3a 42 6f 64 79 3e 3c  70 3a 43 72 65 61 74 65   s:Body>< p:Create
0320  5f 4f 55 54 50 55 54 20  78 6d 6c 6e 73 3a 78 73   _OUTPUT  xmlns:xs
0330  69 3d 22 68 74 74 70 3a  2f 2f 77 77 77 2e 77 33   i="http: //www.w3
0340  2e 6f 72 67 2f 32 30 30  31 2f 58 4d 4c 53 63 68   .org/200 1/XMLSch
0350  65 6d 61 2d 69 6e 73 74  61 6e 63 65 22 20 78 6d   ema-inst ance" xm
0360  6c 6e 73 3a 70 3d 22 68  74 74 70 3a 2f 2f 73 63   lns:p="h ttp://sc
0370  68 65 6d 61 73 2e 6d 69  63 72 6f 73 6f 66 74 2e   hemas.mi crosoft.
0380  63 6f 6d 2f 77 62 65 6d  2f 77 73 6d 61 6e 2f 31   com/wbem /wsman/1
0390  2f 77 6d 69 2f 72 6f 6f  74 2f 63 69 6d 76 32 2f   /wmi/roo t/cimv2/
03a0  57 69 6e 33 32 5f 50 72  6f 63 65 73 73 22 20 78   Win32_Pr ocess" x
03b0  6d 6c 6e 73 3a 63 69 6d  3d 22 68 74 74 70 3a 2f   mlns:cim ="http:/
03c0  2f 73 63 68 65 6d 61 73  2e 64 6d 74 66 2e 6f 72   /schemas .dmtf.or
03d0  67 2f 77 62 65 6d 2f 77  73 63 69 6d 2f 31 2f 63   g/wbem/w scim/1/c
03e0  6f 6d 6d 6f 6e 22 3e 3c  70 3a 50 72 6f 63 65 73   ommon"><  p:Proces
03f0  73 49 64 3e 31 37 38 38  3c 2f 70 3a 50 72 6f 63   sId>1788 </p:Proc
0400  65 73 73 49 64 3e 3c 70  3a 52 65 74 75 72 6e 56   essId><p :ReturnV
0410  61 6c 75 65 3e 30 3c 2f  70 3a 52 65 74 75 72 6e   alue>0</ p:Return
0420  56 61 6c 75 65 3e 3c 2f  70 3a 43 72 65 61 74 65   Value></ p:Create
0430  5f 4f 55 54 50 55 54 3e  3c 2f 73 3a 42 6f 64 79    OUTPUT> </s:Body
```

Packet Details pane showing decrypted WSMan XML traffic and the calculator application ProcessID.

14. https://docs.extrahop.com/current/ssl-decryption-concepts/ssl-decryption-concepts.pdf

Adversarial LOLBAS Usage

In the current threat landscape, the use of PowerShell via WMI and PSRP over WinRM remains a staple LOLBAS technique for lateral movement and persistence. This method continues to be highly favored by state-sponsored actors because it utilizes legitimate system tools to blend in with administrative traffic. A couple of notable APT groups have been identified in recent reports, between 2025 and 2026, utilizing these specific tactics:

- State-sponsored Chinese threat actors, including groups like Volt Typhoon[15] and APT 41,[16] use this technique to achieve objectives, including intellectual property theft, corporate espionage, and critical infrastructure compromise.[17] The 2023 Volt Typhoon campaign[18] compromised U.S. critical infrastructure, maintaining months of persistent access without traditional malware by using only legitimate Windows tools, remote utilities, and self-signed certificates. This campaign was highly successful because the operation stayed within trusted, rarely scrutinized system boundaries.

- State-sponsored Iranian threat actors, including MuddyWater,[19] frequently utilize WMI and PowerShell to execute code on remote systems (**T1047**). Recent campaigns reported on March 6, 2026, involving "pre-planted backdoors" in Gulf state infrastructure have highlighted their continued reliance on LOLBAS for stealthy persistence.

The Visibility Gap: NDR vs. EDR/Logging

The core of the lateral movement problem lies in the "Visibility Triad," where EDR, IDPS, and NDR must work in concert. However, in the context of native architecture and protocols, EDR exhibits several fundamental deficiencies that NDR is uniquely positioned to address.[20]

EDR Blind Spots: Unmanaged Devices and Log Tampering

EDR platforms are fundamentally restricted to the hosts on which they are installed. In modern hybrid environments, this leaves significant blind spots:

- **Unmanaged Assets:** IoT devices, legacy servers, and "Bring Your Own Device" (BYOD) endpoints cannot host EDR agents.[21] An attacker pivoting from a compromised laptop to an unmanaged printer or a legacy industrial controller is invisible to EDR.

- **Log Suppression:** Threat actors often begin their post-exploitation phase by disabling Event Tracing for Windows (ETW) or clearing Security and System logs. If the logs are deleted before they are forwarded to a SIEM, the forensic trail is lost.

- **AMSI Bypasses:** PowerShell's Anti-Malware Scan Interface (AMSI) is the primary "hook" used by EDR to inspect script blocks. Attackers frequently use memory patching (e.g., AmsiScanBuffer patching) to disable this hook, rendering the EDR blind to the malicious script content.[22]

15. https://www.intel471.com/blog/hunting-apts-from-state-policy-to-ttps
16. https://www.cyfirma.com/research/apt-profile-mission2025/
17. https://industrialcyber.co/ransomware/mission2025-cyber-campaign-expands-global-targeting-of-manufacturing-critical-infrastructure/
18. https://www.cisa.gov/news-events/cybersecurity-advisories/aa24-038a
19. https://socradar.io/blog/cyber-reflections-us-israel-iran-war/
20. https://cloud-assets.extrahop.com/resources/papers/myth-ndr-datasheet.pdf
21. https://www.sentinelone.com/cybersecurity-101/endpoint-security/ndr-vs-edr/
22. https://learn.microsoft.com/en-us/defender-endpoint/amsi-on-mdav

Gaining Insight into Unauthorized Activities with Decryption

ExtraHop provides an immutable record of network behavior that cannot be tampered with by an adversary on a local host. For detecting lateral movement, having an advanced NDR is crucial:

- **Protocol-Level "Ground Truth":** Advanced NDR tools provide visibility by decrypting RPC payloads and SOAP headers in WSMAN traffic. It is necessary to see the decrypted commands sent to victim computers to fully characterize an intrusion. Consequently, analyzing the behavioral patterns of these flows, such as looking at anomalies in "East-West" traffic volume, often provides an earlier indication of a threat post-compromise than host-based alerts.
- **Network Decryption:** By integrating with AD and forwarding TLS session keys, these solutions can decrypt protocols like WSMAN (when encrypted with Kerberos, NTLM, or TLS) and RPC (either encrypted directly over TCP or when encrypted via SMB). This action is essential for defenders, as it reveals the true substance of lateral movement, even when attackers employ encryption, providing visibility into transferred files, executed commands, and tested credentials.
- **Contextual Awareness:** NDR excels at identifying unmanaged devices and linking their network activity to managed assets. For instance, an NDR can detect and alert on an unmanaged device initiating a series of WMI queries toward a domain controller, a behavioral activity that EDR solutions will overlook.

MITRE ATT&CK Mappings

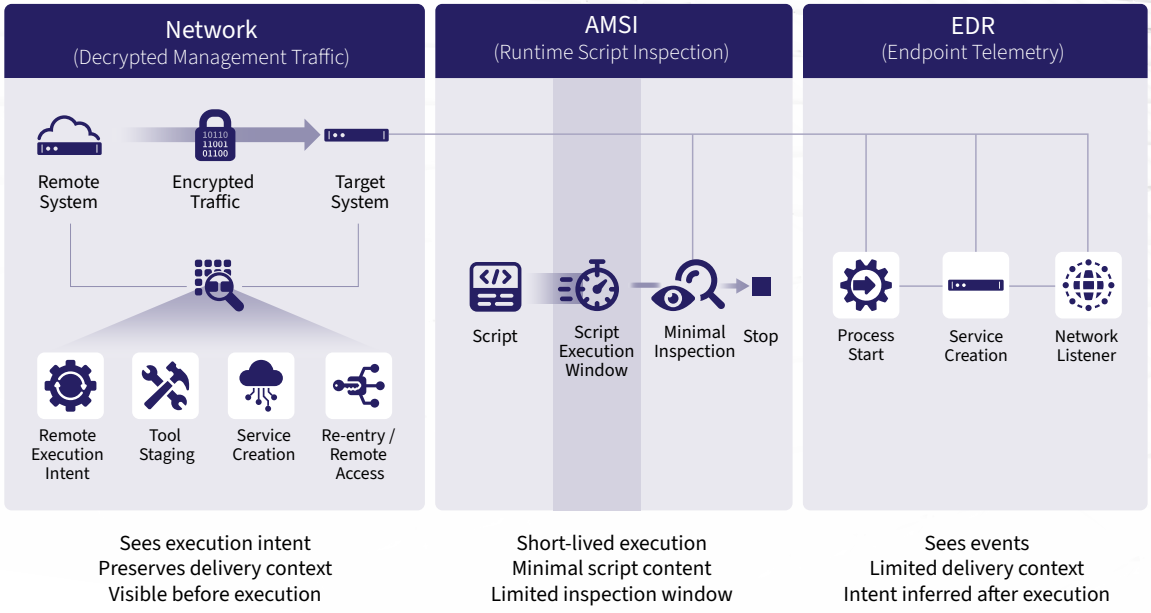
To effectively counter the sophisticated use of LOLBAS techniques, security teams must align their visibility strategies with industry-standard frameworks. The following table maps common lateral movement activities involving WMI, PowerShell, and WinRM/WSMAN to the MITRE ATT&CK framework.

Table 2: MITRE ATT&CK Mapping and WMI over WSMAN and RPC Detections

MITRE ATT&CK ID	Tool/Protocol/Service	Threat Actor Activity	Detection
T1047	WMI Enumeration	Actor sends enumeration queries through WMI	New WMI Enumeration Query
T1047	Windows Management	Actor uses WMI to execute malicious commands and payloads remotely	New WMI Process Creation Activity
T1021.006	WinRM/WSMAN	Attacker sends commands over WSMAN using a valid account	New WSMan Remote Administration Activity Inspect SOAP envelopes for **New-PSSession or Invoke-Command** strings[23]
T1543.003	Create or Modify System processes	Actor sends a request to launch a service, including LOLBAS	New Remote Service Launch Attempt Remote Service Launch Attempt to Run a LOLBAS
T1059.001	PowerShell, including PowerShell over WSMAN	Actor issues native PowerShell commands such as **'Get-Service'** and **'Invoke-Expression'**	New PowerShell Remoting Attempt

23. https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/new-pssession?view=powershell-7.5

Conclusion



NDR decryption vs AMSI vs EDR

The strategic abuse of PowerShell over WMI and WSMAN will remain a primary vector for lateral movement as long as Windows remains a backbone of the modern enterprise. The paradox of these protocols is that their utility for administrators is precisely what makes them particularly lethal in the hands of an adversary. For technical professionals, the defense must shift from simple port monitoring to deep, decrypted protocol analysis. Metadata is not enough in an era where the most sophisticated attacks use "trusted" channels. Decryption is the only way to distinguish a legitimate administrator's functions from a threat actor's lateral movement.

By turning the network into a source of ground truth, organizations can cut through the mask of encryption, shrink attacker dwell time, and contain threats before they escalate into material breaches. In the modern data center, visibility is the only true defense against the enemies that are already inside.

ABOUT EXTRAHOP

ExtraHop turns the network—the enterprise's ultimate source of truth—into actionable insight to power security, performance, and resilience. Delivering superior data by design, we ensure superior defense by default.

The ExtraHop modern network detection and response (NDR) platform provides visibility that thinks, analyzing behavior to intercept evasive risks before they cause damage. We transform network noise into definitive context, enabling security teams to make faster decisions and operate at uncompromised scale.

Whether securing cloud modernization or de-risking AI adoption, ExtraHop gives global enterprises the ground truth they need to thrive.

To learn more, visit extrahop.com or follow us on LinkedIn.

EXTRAHOP®

info@extrahop.com
extrahop.com