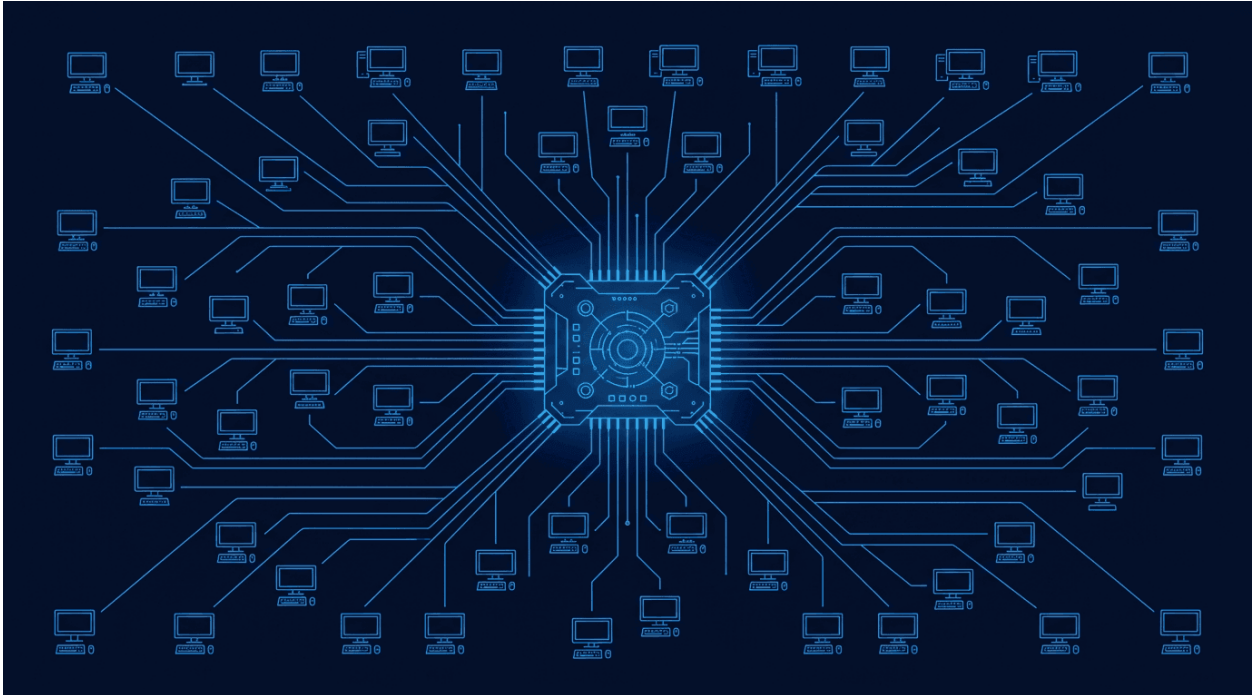


Ansible Alternative: Why I Use Action1 for Windows Management

Tue, May 5, 8:15 AM EDT · <https://scottslab.io/posts/action1-windows-management-ansible-alternative>



TL;DR

Ansible is good at Linux and miserable at Windows, so the Windows endpoints go to Action1 instead. Agent install runs about two minutes a box against hours of WinRM wrangling, the free tier covers 200 endpoints, and the agent dials out. No inbound rules to open. It does not replace Ansible; the two split the fleet by operating system.

Published: May 2026 **Category:** Infrastructure Automation **Reading Time:** 12 minutes

Executive Summary

- Deployed Action1 RMM for Windows endpoint management - free tier covers 200 endpoints (more than enough for home lab)
- Agent installation takes under 2 minutes per endpoint vs. hours of WinRM configuration for Ansible
- Cloud-based console provides patch management, software deployment, remote scripting, and real-time monitoring
- Complements Ansible rather than replacing it: Action1 handles Windows, Ansible handles Linux

- No inbound firewall rules required - agent initiates outbound connection to Action1 cloud
-

Goal

Problem: Ansible is fantastic for Linux automation, but Windows support has always been... friction-heavy. WinRM configuration, certificate management, PowerShell execution policies, firewall rules, credential delegation - every Windows host requires significant prep work before Ansible can touch it. For a home lab with a handful of Windows machines (gaming PC, media center, occasional test VMs), that overhead isn't worth it.

Why it mattered: I still need to manage Windows endpoints: push updates, deploy software, run scripts, check security posture. I wasn't going to RDP into each machine individually like it's 2005. I needed something purpose-built for Windows that "just works" - and ideally doesn't cost anything for home lab scale.

Scope and Constraints

In Scope

- Windows endpoint management (patching, software deployment, scripting)
- Agent deployment to home Windows machines
- Patch compliance monitoring
- Software inventory and deployment
- Remote PowerShell execution

Out of Scope

- Linux management (Ansible handles this)
- Enterprise features (compliance reporting, advanced RBAC)
- On-premises management server (Action1 is cloud-only)
- Integration with existing Ansible workflows

Key Constraints

- **Cost** - Must be free for home lab use (200 endpoint limit is fine)
 - **Complexity** - Must be simpler than configuring WinRM for Ansible
 - **Security** - Agent must not require inbound firewall rules
 - **Coverage** - Must handle core RMM tasks: patching, software, scripting
-

Tools and References

Tool	Role in This Project
Action1	Cloud-based RMM platform for Windows endpoint management
Action1 Agent	Lightweight Windows service (~15MB) that connects to cloud console
PowerShell	Scripting engine for custom automation via Action1
Windows Update	Patch source managed through Action1 policies

References:

- [Action1 Free Tier](#)
 - [Action1 Documentation](#)
 - [Why WinRM is Painful](#)
-

Approach

Phase 1: Evaluate the Windows Automation Problem

What I did: Attempted to configure Ansible for Windows management. Set up WinRM listeners, configured HTTPS with self-signed certificates, adjusted PowerShell execution policies, opened firewall ports, configured CredSSP for credential delegation.

Why I stopped: After 3 hours on the first Windows host, I still had intermittent authentication failures. The juice wasn't worth the squeeze for 4 Windows endpoints.

Phase 2: Evaluate RMM Alternatives

What I did: Researched RMM (Remote Monitoring and Management) tools with free tiers: Action1, Tactical RMM, MeshCentral, PDQ (limited free). Evaluated based on: free tier limits, ease of setup, feature set, security model.

Why Action1: 200 free endpoints (way more than I need), cloud-hosted (no server to maintain), agent-initiated connections (no inbound firewall rules), and solid feature set (patching, software deployment, scripting, inventory).

Phase 3: Agent Deployment

What I did: Downloaded the Action1 agent MSI from the cloud console. Installed on Windows endpoints via simple MSI execution - no configuration required. Agent auto-registered with my Action1 organization.

Why this matters: Compare "run MSI, done" to "configure WinRM listener, generate certificate, configure firewall, set execution policy, test connectivity, debug authentication errors." The time savings are significant.

Phase 4: Configure Patch Management

What I did: Created patch policies in Action1 console: approve critical/security updates automatically, schedule weekly patch scans, configure maintenance windows for automatic installation.

Why: Unpatched Windows machines are a liability. Automated patching ensures endpoints stay current without manual intervention.

Phase 5: Software Deployment Setup

What I did: Built software deployment packages for common applications: browsers, utilities, security tools. Action1 has a built-in repository of common software plus support for custom packages.

Why: When I rebuild a Windows machine, I don't want to manually download and install 15 applications. Deploy from Action1, wait 10 minutes, done.

Phase 6: Remote Scripting Configuration

What I did: Created PowerShell script library in Action1 for common tasks: system information gathering, security configuration checks, cleanup scripts. Scripts run as SYSTEM with full privileges.

Why: Sometimes you need to run a quick command across all Windows machines. Action1's script execution is faster than RDPing into each one.

Implementation Notes

Agent Installation (Sanitized)

```
# Download agent MSI from Action1 console (URL is organization-specific)
# URL format: https://app.action1.com/agent/download/<ORG_ID>

# Silent installation
msiexec /i Action1Agent.msi /qn

# Agent auto-registers with Action1 cloud
# No additional configuration required
```

That's it. No WinRM configuration. No certificates. No firewall rules. The agent initiates an outbound HTTPS connection to Action1's cloud - works through NAT, works through firewalls, works everywhere.

Patch Policy Configuration (Sanitized)

Policy: Home Lab Windows Patching

Scan Schedule: Weekly (Sunday 2:00 AM)

Auto-Approve:

- Critical Updates: Yes
- Security Updates: Yes
- Definition Updates: Yes
- Feature Updates: No (manual review)
- Driver Updates: No (manual review)

Installation Window:

- Day: Sunday
- Time: 3:00 AM - 6:00 AM
- Reboot: If required, after window

Endpoints: All Windows endpoints

Software Deployment Example (Sanitized)

Package: Firefox Browser

Source: Action1 Repository (pre-built)

Version: Latest

Installation: Silent (/S flag)

Detection: Registry check for installed version

Deployment:

- Target: All Windows endpoints
- Schedule: Immediate
- Retry on failure: 3 attempts

Custom PowerShell Script Example (Sanitized)

```
# Script: Get-SecurityStatus.ps1
# Purpose: Check Windows security configuration

$results = @{
    Hostname = $env:COMPUTERNAME
    WindowsFirewall = (Get-NetFirewallProfile | Where-Object {$_.Enabled -eq $true}).Name
    DefenderStatus = (Get-MpComputerStatus).AntivirusEnabled
    LastUpdate = (Get-HotFix | Sort-Object InstalledOn -Descending | Select-Object -First 1).InstalledOn
    UAC = (Get-ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System).EnableLUA
}

$results | ConvertTo-Json
```

Ansible vs Action1 Comparison

Task	Ansible (Windows)	Action1
Initial setup per host	30-60 minutes (WinRM, certs, firewall)	2 minutes (run MSI)
Patch management	Custom playbook + WSUS or direct	Built-in policy engine

Task	Ansible (Windows)	Action1
Software deployment	win_package module + hosting	Built-in repository
Remote scripting	win_shell module	Web console or API
Agentless	Yes (but requires WinRM)	No (lightweight agent)
Works through NAT	Requires port forwarding	Yes (outbound only)
Cost	Free	Free (200 endpoints)

Validation and Evidence

Signals That Proved It Worked

Check	Expected	Actual
Agent registration	Endpoints visible in console	All 4 Windows endpoints registered
Patch scan	Compliance report generated	Shows missing patches per endpoint
Software deployment	Firefox installed silently	Deployed to all endpoints in <5 min
Remote script	JSON output returned	Security status collected from all
Patch installation	Updates installed overnight	Confirmed via compliance dashboard

Validation Steps (Sanitized)

- Action1 Console → Endpoints
 - Verify all Windows machines appear
 - Check "Last Seen" timestamp (should be recent)
- Action1 Console → Patch Management → Compliance
 - Verify scan completed
 - Review missing patches
- Action1 Console → Apps → Managed Apps
 - Verify deployed software shows "Installed" status
- Action1 Console → Scripts → Run Script
 - Execute test script
 - Verify output returned from all endpoints

Results

Metric	Outcome
Windows Endpoints Managed	4 (gaming PC, media center, 2 test VMs)
Setup Time per Endpoint	~2 minutes (MSI install)
Patch Compliance	Automated weekly scan + install
Software Packages	8 apps deployed via Action1
Custom Scripts	5 PowerShell scripts in library
Cost	\$0 (free tier, 200 endpoint limit)
Ansible WinRM Configs Avoided	4 (and counting)

What I Learned

1. **The right tool for the job beats forcing a universal tool.** Ansible is excellent for Linux. Fighting with WinRM to make it work on Windows isn't worth it when purpose-built tools exist.
2. **Agent-based beats agentless for Windows in home labs.** Ansible's agentless model requires WinRM configuration on every endpoint. Action1's lightweight agent (15MB, runs as service) requires zero endpoint configuration.
3. **Outbound-only connections simplify everything.** No inbound firewall rules, no port forwarding, no NAT traversal issues. The agent calls home; the cloud console is always reachable.
4. **200 free endpoints is generous for home use.** I have 4 Windows machines. Even if I had 20, I'd still be well under the limit. Enterprise features are paid, but home lab features are free.
5. **Cloud-hosted RMM means no infrastructure to maintain.** No server to patch, no database to back up, no certificates to renew. Trade-off: you're trusting a third party with endpoint access.
6. **Built-in patch management is a massive time saver.** Writing Ansible playbooks for Windows Update is tedious. Action1's policy engine handles it with a few clicks.
7. **Software repository with common apps eliminates hosting.** I don't need to maintain an internal package repository for Firefox, Chrome, 7-Zip, etc. Action1 has them ready to

deploy.

8. **PowerShell scripts run as SYSTEM.** Full privileges, no UAC prompts, consistent execution environment. More powerful than what you'd get interactively.
 9. **Complements Ansible rather than replacing it.** My Linux hosts still use Ansible. My Windows hosts use Action1. Different tools, same goal: automated management.
 10. **Time saved on setup is time spent on actual work.** The hours I didn't spend debugging WinRM went into building detection rules, writing playbooks, and improving actual security posture.
-

What I Would Improve Next

P0 (Do This Week)

- **Endpoint naming standardization** - Rename endpoints in Action1 to match inventory naming convention
- **Script library expansion** - Add scripts for common troubleshooting tasks

P1 (Do This Month)

- **Automated onboarding** - Create deployment script that installs Action1 agent as part of Windows setup
- **Security baseline policy** - Build policy that checks/enforces security settings (firewall, UAC, Defender)
- **Alert configuration** - Set up alerts for offline endpoints and failed patch installations

P2 (Do This Quarter)

- **Integration with monitoring** - Export Action1 events to SIEM for unified visibility
 - **Custom package repository** - Add internal applications to Action1 for deployment
 - **Endpoint groups** - Organize endpoints by function (workstations, servers, test VMs)
 - **Reporting automation** - Schedule weekly compliance reports via email
-

Common Failure Modes

1. **"Endpoint shows offline in console"** - Check if Windows machine is powered on and has internet connectivity. Agent requires outbound HTTPS to Action1 cloud.

2. **"Patch installation failed"** - Check available disk space (Windows Update needs room). Review Action1 logs for specific error codes. May need to clear Windows Update cache.
 3. **"Software deployment stuck at 'Pending'"** - Endpoint may be offline or agent service may have stopped. Restart Action1 Agent service on the endpoint.
 4. **"Script execution returned no output"** - Script may have errored. Check script syntax in PowerShell ISE locally first. Action1 captures stderr but display can be confusing.
 5. **"Can't find endpoint after reinstalling Windows"** - Old endpoint entry is orphaned. Delete the old entry in Action1 console, reinstall agent on fresh Windows install.
-

Security Considerations

Agent Security Model

- Agent runs as SYSTEM service (high privilege, but contained to endpoint)
- Communication is outbound HTTPS only (no inbound attack surface)
- Agent authenticated to Action1 org via unique endpoint ID
- All management traffic encrypted in transit

Cloud Trust Model

- Action1 is a third-party SaaS - you're trusting them with endpoint access
- They can execute scripts, install software, access file systems on managed endpoints
- Appropriate for home lab; evaluate carefully for business use
- SOC 2 Type II certified (for those who care about compliance)

Access Control

- Action1 console protected by username/password + optional MFA
- Role-based access available (not needed for single-user home lab)
- Audit log tracks all actions taken through console

Comparison to Ansible Security

- Ansible: credentials stored on control node, WinRM requires authentication config
 - Action1: credentials managed by SaaS, agent pre-authenticated at install
 - Both require trust in the management platform; different trust models
-

Runbook

How to Add a New Windows Endpoint

1. Log into Action1 console
2. Go to Endpoints → Add Endpoints
3. Download agent MSI (or use existing downloaded installer)
4. On Windows endpoint: run `msiexec /i Action1Agent.msi /qn`
5. Wait 1-2 minutes for endpoint to appear in console
6. Verify endpoint shows "Online" status

How to Deploy Software to All Windows Endpoints

1. Action1 Console → Apps → Deploy App
2. Select from repository or upload custom package
3. Choose target: All Endpoints (or specific group)
4. Configure installation options (silent, reboot behavior)
5. Click Deploy
6. Monitor progress in deployment status view

How to Run a Script Across All Endpoints

1. Action1 Console → Scripts → Run Script
2. Select existing script or create new
3. Choose target endpoints
4. Click Run
5. View output in script execution results

How to Check Patch Compliance

1. Action1 Console → Patch Management → Overview
2. Review compliance percentage per endpoint
3. Click endpoint for detailed missing patch list
4. Approve/deploy patches manually or wait for policy

How to Troubleshoot Offline Endpoint

1. Verify Windows machine is powered on
2. Check internet connectivity on the endpoint
3. Open Services (services.msc), find "Action1 Agent"
4. If stopped, start the service

5. If running, restart the service
6. Check Action1 console in 1-2 minutes for online status

Appendix

Glossary

Term	Definition
RMM	Remote Monitoring and Management - tools for managing endpoints remotely
Action1	Cloud-based RMM platform with free tier for up to 200 endpoints
WinRM	Windows Remote Management - Microsoft's implementation of WS-Management protocol
Agent	Software installed on endpoint that communicates with management platform
Agentless	Management approach that doesn't require software on managed endpoints
SYSTEM	Windows built-in account with highest local privileges
MSI	Microsoft Installer package format for Windows software deployment
Patch Policy	Rules defining how and when updates are approved and installed

Why Not Just Use Ansible for Windows?

Ansible Windows Challenge	Impact
WinRM listener configuration	15-30 min per host
Certificate management (HTTPS)	Ongoing maintenance
PowerShell execution policy	Must be configured permissively
Firewall rules	Inbound ports required
CredSSP for delegation	Security implications, complex setup
Authentication debugging	"It should work but doesn't" syndrome
NAT traversal	Requires port forwarding or VPN

Assumption: For enterprise environments with hundreds of Windows servers and existing WinRM infrastructure, Ansible makes sense. For home labs with a handful of Windows endpoints, the setup cost isn't justified.

Action1 Free Tier Limitations

Feature	Free Tier	Paid Tier
Endpoints	200	Unlimited
Patch Management	Yes	Yes
Software Deployment	Yes	Yes
Remote Scripting	Yes	Yes
Reporting	Basic	Advanced
RBAC	Limited	Full
API Access	Limited	Full
Support	Community	Priority

Artifacts Produced

- **Action1 Organization** - Cloud-hosted management console for Windows endpoints
- **Endpoint Agents** - Action1 agent deployed to 4 Windows machines
- **Patch Policy: Home Lab Windows** - Weekly scan, auto-approve security updates, Sunday maintenance window
- **Software Packages** - 8 applications configured for deployment (browsers, utilities, tools)
- **Script Library** - 5 PowerShell scripts for common management tasks
- **Documentation: Ansible vs Action1** - Decision record for tool selection

Still fighting with Ansible and Windows? Give Action1 a try. The free tier is genuinely free, and you might actually finish setting it up.

Written by Scott S. — Contact: scott@scottschlangen.com — scottslab.io — <https://scottslab.io/posts/action1-windows-management-ansible-alternative>