
WHITE PAPER

# A System Hardening Checklist for Managing Security at Every Layer

## The Decision Maker's Guide to Hardening Systems & Staying Compliant

## Introduction

System hardening is a way to conceptualize how resilient your IT systems are against attacks and vulnerabilities. While there's no single approach to system hardening, the increasing sophistication of attacks and the proliferation of software vulnerabilities means managing system hardening manually just isn't viable for IT of any scale.

This checklist for decision makers on IT teams explains the primary ways to harden systems at key layers, from the server to the application and beyond, and best practices for building a long-term, continuously enforceable system hardening strategy using automation.

# The Importance of Building Resilient IT for Security, Compliance & Trust

There are two main reasons IT system hardening has become a crucial aspect of cybersecurity. First is the increasing sprawl of the digital landscape: Apps, workflows, and data live in more places than ever (data center, cloud, hybrid, edge, and beyond), increasing availability and exposure at the same time.

Which leads us to the second reason: An endless barrage of increasingly sophisticated cyber threats from every direction, only becoming more treacherous every year. And attack is just one category of threat, complemented by an unprecedented wave of software vulnerabilities cropping up with each patch, update, and tool you add to your stack.

> "We're on a trajectory to have the most vulnerabilities ever identified in a single year, starting this year."
>
> Sean Atkinson, CISO, The Center for Internet Security, 2024
>
> Pulling the Strings Season 6, Episode 3: "When Will IT Security Escape the Cat-and-Mouse Game?"

Encompassing everything from login protections to firewalls and more, it's easy to see system hardening as just a big, broad shield between your critical systems and the risks of today's chaotic digital reality (including heavy fines and brand damage). But the value and significance of hardening extends beyond mere security and compliance; it also plays a vital role in **building and maintaining resilient IT that supports your business goals,** regardless of your industry or which technologies you choose for your stack.

Regulations like GDPR, HIPAA, CMMC 2.0, and PCI DSS require organizations to apply patches, perform recurring assessments, and implement robust security controls — almost all aligned with the principles of IT system hardening. These requirements often apply across a patchwork of environments: Windows, Linux, on-prem, cloud, and beyond.

As such, hardening becomes a common initiative that spans different regulations, frameworks, and technologies, helping to **simplify what often seems like a daunting effort and unlock greater value from your IT by leveraging your preferred tools and workflows.** This is especially true in a large IT environment, where your choice of enterprise operating systems like Linux distributions should match what your organization's business priorities are — not held back by the system hardening controls you can enforce across OSes.

Even beyond security and compliance, hardening has become a trust-building differentiator. Trust is essential in today's competitive market: Customers, partners, and investors are increasingly concerned about data privacy and security. Demonstrating a commitment to hardening IT systems can enhance your organization's reputation, signaling that you take cybersecurity seriously.

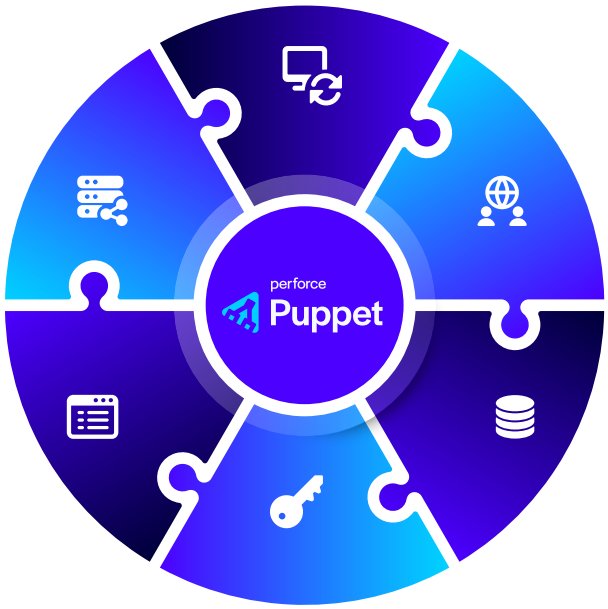
In this guide, you'll see the breadth, complexity, and detail of what we call 'system hardening.' Creating and enforcing a system hardening strategy built around reliable, accessible measures like these is core to a proactive, future-focused IT security and compliance strategy. They let your organization reduce vulnerabilities, adhere to regulatory requirements, and safeguard digital assets, ensuring security, compliance, and trust — pillars of what it means to build resilient IT today.

# A System Hardening Checklist for IT Decision Makers

System hardening doesn't just mean protecting an IT system against vulnerabilities and attacks. More specifically, it means securing **system components according to an industry standard and keeping them there,** rather than arbitrarily decreeing that your system is "secure" or that your users and devices are "trusted."

System hardening is useful because it gives you a measuring stick against which to judge the effectiveness and appropriateness of your cybersecurity tools, protocols, and practices. Aligning system security to an established standard for hardening aligns your IT system security to a best-practice state that can be audited, enforced, and updated.

Imagine a sliding puzzle where each tile represents the security configurations and practices for each component of your IT system: One for server security, one for the operating system (OS), one for applications, network, database, and so on.




Only once you line them up properly can you reasonably consider the system "secure." Otherwise, it's an incomplete portrait, or it just doesn't make sense in the larger context. **System hardening is like having a picture of the completed puzzle**. You might need to do some things differently to accommodate your particular pieces (your deployment platforms, OSes, and other elements of your tech stack), but you can see how each piece needs to fit together to make it work.

And when the "ideal" puzzle changes — like when security standards evolve, a new regulation appears, or you replace elements of your tech stack — you must adjust the pieces to keep security and compliance in focus.

Each page in this section is dedicated to a different element or type of system hardening. Read on for a checklist of cybersecurity measures, protocols, and practices you should be implementing and enforcing to ensure system hardening across your critical infrastructure.

## Notes:

- **Many system hardening measures apply to more than one specific kind of hardening.** For example, patching and updating is important both at the OS and the application level. Thus, we refer to them broadly as "system hardening measures," though they may also be referenced in the individual hardening sections where they apply.
- **Most compliance directives, certifications, and regulations call for a broad range of common system hardening measures.** For example, PCI DSS, CMMC 2.0, and ISO 27001 are just a few that call for the use of multi-factor authentication (MFA) in the identity management portion of the security policy. For each item on the checklist, we've highlighted select compliance expectations that call for that checklist item.

## Server Hardening

**Securing components and permissions of hardware, software, and firmware layers of a system. Can include patching, updating, multi-factor authentication (MFA), and strong password use.**

- **Primary purpose:** Protecting hosted applications & services, databases, auth, and critical workloads
- **Key measures:** MFA, RBAC, disabling unnecessary services
- **Required by:** NIST, PCI DSS, ISO 27001 & more

### Require strong passwords

- ☐ Windows Group Policy
- ☐ Pluggable Authentication Module password policies (Linux)

### Enable multi-factor authentication (MFA)

- ☐ Okta
- ☐ Authy
- ☐ Google Authenticator
- ☐ Microsoft Authenticator
- ☐ Cisco DUO
- ☐ LastPass

### Use role-based access control (RBAC) to restrict server access

- ☐ sudo (Linux/Unix)
- ☐ Active Directory (Windows)
- ☐ HashiCorp Vault

### Disable or remove guest, default, or inactive accounts

- ☐ Puppet Enterprise Advanced
- ☐ PowerShell
- ☐ Centrify

### Enforce secure configuration baselines

- ☐ Puppet Enterprise Advanced (includes built-in CIS-CAT® Pro Assessor for CIS
- ☐ Benchmark assessment)
- ☐ OpenSCAP (Linux scanner)

### Disable root login via SSH

- ☐ Automation scripts
- ☐ Puppet (for managing SSH config across systems)
- ☐ OpenSSH settings (Windows)

## Operating System Hardening

Securing the software that manages the hardware and grants server permissions to application software. Usually handled with automatic updates and patches, but can also include tasks like removing unnecessary drivers, terminating unneeded services, limiting user creation, encrypting HDDs/SSDs, and more.

- **Primary purpose:** Protecting the system kernel, file system integrity, and user accounts
- **Key measures:** Applying security patches, configuring firewalls, restricting user permissions, disabling unused features & enforcing kernel-level protections
- **Required by:** ISO 27001, SOC 2, NIST SP 800-171, HIPAA & more

### Kernel hardening (Linux)

- ☐ SELinux
- ☐ Grsecurity
- ☐ Mandatory Access Control (MAC)
- ☐ SELinux
- ☐ AppArmor

### Enforce secure OS baselines like CIS Benchmarks and DISA STIGs

- ☐ Puppet Enterprise Advanced policy as code (includes built-in CIS-CAT® Pro Assessor and automated enforcement)

### Configure secure OS kernel booting

- ☐ UEFI Secure Boot (Linux)
- ☐ GRUB (Linux)
- ☐ Microsoft BitLocker (Windows)

### Patch and update OS regularly

- ☐ WSUS (Windows)
- ☐ yum (Linux package manager)
- ☐ Chocolatey (Windows package manager)

### Restrict permissions for OS directories

- ☐ chmod/chown (Linux)
- ☐ PowerShell (Windows)

## Network Hardening

Configuring network firewalls, disabling services, auditing access privileges, encrypting traffic, and more. Often includes intrusion prevention and detection software to monitor suspicious activity and prevent unauthorized network access.

- **Primary purpose:** Protecting data in transit, ports and interfaces, and internal/external network access
- **Key measures:** Configuring firewalls, enforcing encryption protocols, securing open ports & isolating networks
- **Required by:** GDPR, PCI DSS, ISO 27001, CMMC 2.0, NIST SP 800-53 & more

### Configure firewalls to control traffic

- ☐ iptables/nftables (Linux)
- ☐ Windows Defender (Windows)
- ☐ Palo Alto Networks Next-Generation Firewall (NGFW)

### Configure secure DNS

- ☐ Domain Name System Security Extensions (DNSSEC)
- ☐ Cloudflare DNS
- ☐ Microsoft DNS Server (Windows)

### Disable unused ports

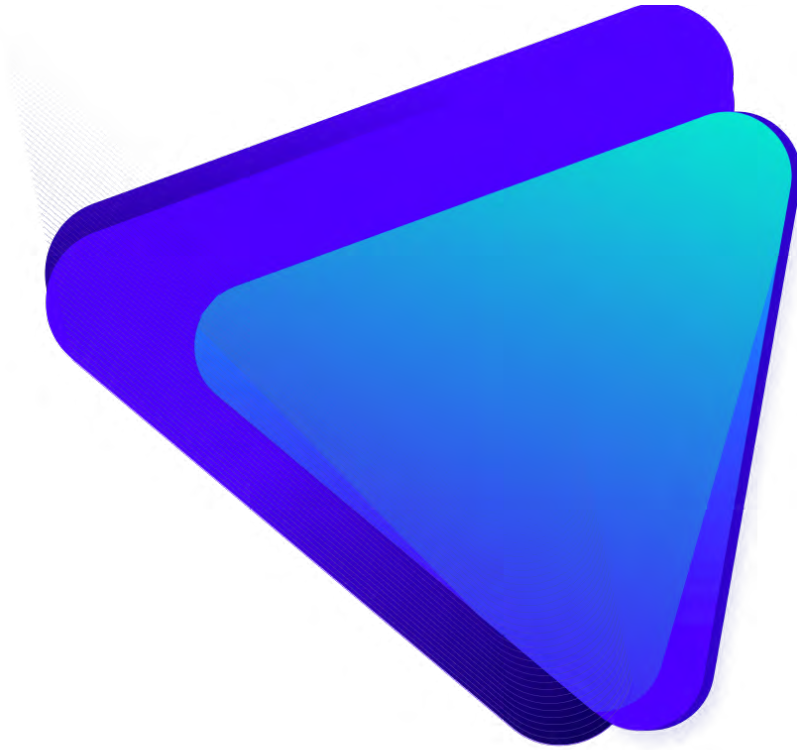
- ☐ netstat/lsof (Linux)
- ☐ PowerShell (Windows)
- ☐ FirewallD (Linux)

### Enable secure communication protocols

- ☐ OpenSSH (Linux)
- ☐ Let's Encrypt
- ☐ Apache HTTP Server
- ☐ HAProxy

### Segregate critical systems with VLANs to limit lateral movement

- ☐ Cisco Catalyst Switches
- ☐ VMware NSX
- ☐ AWS Virtual Private Cloud (VPC)



## Application Hardening

Patching application code, using antivirus software, encrypting and managing passwords, and using firewalls to secure a server's applications.

- **Primary purpose:** Protecting app-specific configurations, runtime, user input, and in-app authentication
- **Key measures:** Disabling unnecessary features, securing APIs, applying patches, enforcing input validation & managing permissions
- **Required by:** SOX, PCI DSS, OWASP & more

**Disable unnecessary features**

- ☐ Windows Group Policy

**Disable default credentials**

- ☐ Tenable Nessus
- ☐ CyberArk

**Disable debug modes**

- ☐ Linting tools in build pipelines (Jenkins, GitHub actions)

**Run apps with "least privilege"**

- ☐ SELinux/AppArmor (Linux)
- ☐ Microsoft Active Directory (Windows)
- ☐ RBAC
- ☐ sudo (Linux)

**Validate user input (prevent injection attacks by SQL or command injection)**

- ☐ Burp Suite
- ☐ Security frameworks with validation (like Django and Spring Boot)
- ☐ Perforce Klocwork

**Patch and update apps regularly**

- ☐ WSUS (Windows)

**Enforce multi-factor authentication (MFA)**

- ☐ Okta
- ☐ Authy
- ☐ Google Authenticator
- ☐ Microsoft Authenticator
- ☐ Cisco DUO
- ☐ LastPass

**Configure API gateways to enable secure communications between apps**

- ☐ Kong API Gateway
- ☐ NGINX Plus
- ☐ Postman API

**Test for security (SAST and DAST)**

- ☐ Snyk
- ☐ SonarQubs
- ☐ Perforce Klocwork
- ☐ Perforce QAC (C/C++)

## Database Hardening

Controlling database privileges, disabling database functions, and encrypting database information. Includes patching the database management system (DBMS), using role-based access control (RBAC), restricting administrative privileges, and more.

- **Primary purpose:** Protecting structured data storage, data at rest, and query execution
- **Key measures:** Enforcing encryption, managing database roles and permissions, applying patches, auditing access logs & securing query execution
- **Required by:** PCI DSS, GDPR, HIPAA, SOX, NIST SP 800-53 & more

### Disable unused features, ports, and protocols

- ☐ SQL Server Management Studio (SSMS) (Windows)
- ☐ MySQL
- ☐ CIS-CAT® Pro Assessor to assess unnecessary database features

### Remove default accounts and passwords

- ☐ SQL Server Management Studio (SSMS) (Windows)
- ☐ Tenable Nessus
- ☐ MySQL

### Encrypt data at rest and in transit

- ☐ Transparent Data Encryption (TDE) (SQL Server, Oracle, MySQL)
- ☐ OpenSSL
- ☐ HashiCorp Vault
- ☐ Perforce Delphix data masking & virtualization (for development & testing)

### Maintain access and activity logs for audit trails

- ☐ Microsoft SQL Server
- ☐ Oracle Audit Vault
- ☐ Splunk

### Patch and update database software regularly

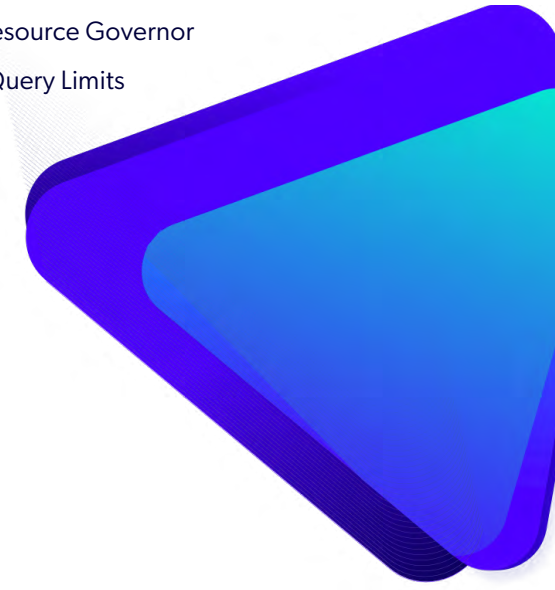
- ☐ WSUS (Windows)
- ☐ Red Hat Satellite
- ☐ AWS Systems Manager Patch Manager (AWS)

### Back up regularly

- ☐ Veeam Backup
- ☐ SQL Server Maintenance Plans
- ☐ AWS Backup

### Enable resource throttling to prevent DoS or DDoS attacks

- ☐ Oracle Resource Manager
- ☐ SQL Server Resource Governor
- ☐ PostgreSQL Query Limits



## Physical Hardening

Preventing access to technology resources in a physical space. Includes intrusion sensors, personnel barriers, and other solutions to harden physical perimeters around system technologies.

- **Primary purpose:** Protecting servers, racks, power, media, switches, and other physical elements of IT
- **Key measures:** Restricting facility access, surveillance, environmental controls, labeling, securing backups & recycling/destroying media securely
- **Required by:** FISMA, PCI DSS, HIPAA & more

**Restrict physical access to systems**

- ☐ Keycards
- ☐ Biometric scanners
- ☐ PIN codes

**Monitor and log physical access**

- ☐ Video surveillance
- ☐ RFID badges
- ☐ Motion sensors

**Prevent unauthorized removal of devices**

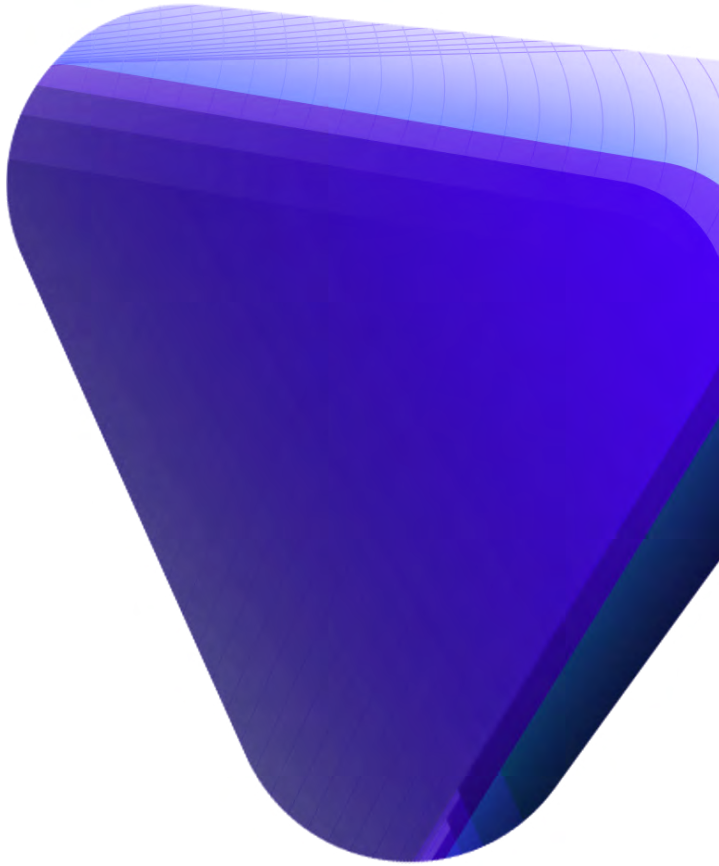
- ☐ Asset management & loss prevention systems
- ☐ RFID tags

**Physically isolate critical systems from non-critical systems to prevent lateral access**

**Use geographically distributed data centers to mitigate risk from attack or natural events**

**Dispose of hardware securely**

- ☐ Wiping/erasure tools
- ☐ Shredders
- ☐ Degaussers
- ☐ Destruction services



# How Puppet Can Maintain Hardening Measures Across All Systems Automatically

Most of the system hardening measures outlined in this white paper can be automated. When we say "automated," we usually mean one of three things:

**Automation scripts written and managed by your engineers and sysadmins**

- PowerShell
- Ruby
- Python
- Bash

**Agentless automation that executes tasks via SSH or WinRM**

- Ansible
- Puppet Bolt

**Agent-based automation that executes tasks locally (i.e., on the system)**

- Puppet Enterprise Advanced
- Chef
- SaltStack

# Always Secure, Always Compliant

Unlike agentless automation tools like Ansible, Puppet automation doesn't need protocols like SSH or WinRM to execute tasks, reducing reliance on the network and securing it against more potential vulnerabilities. In fact, the Puppet agent is one of Puppet's primary strengths, enabling you to **automatically enforce system hardening measures and compliance policies 24/7.**

The Puppet agent runs every 30 minutes by default to make sure the configurations on each server are in line with your coded policies. If they're not, the agent automatically requests a redeployment of configuration code to that server, bringing it right back into compliance to keep it secure.

The last thing you want to have to do once you finally lock down your system hardening measures is go back and fix them.

**But configuration drift happens to everyone.**

A developer changes something they shouldn't have; a surprise update screws up a configuration; a patch changes a dependency that affects other configurations down the line.

Before you know it, you're out of compliance, and you've got to go assess and tweak every system.

**And even when it's not drift…**

… like if you want to update your security configurations, your compliance policy should be the source of truth for your system hardening efforts.

You go to great lengths to build your compliance policies from industry standards and customize them to your business needs.

**Why wouldn't your automation tools keep you aligned to that?**

Agent-based automation like Puppet's enables continuous compliance, which means keeping your configurations in line all across your systems, all the time. When you use agent-based automation to enforce compliant system configurations, you can:

- **Leverage established compliance frameworks like CIS Benchmarks, DISA STIGs, and NIST** to build a compliance policy
- **Define a compliance policy that's specific to your organization** (with custom rules and exceptions)
- **Stay secure and compliant with 48 agent runs per day,** continuously fixing drift and enforcing your compliance policies
- **Deploy your compliance policy across different OSes,** including Windows, many Linux distributions, and macOS

## Control it All from One Place, No Matter Where Your Systems Live

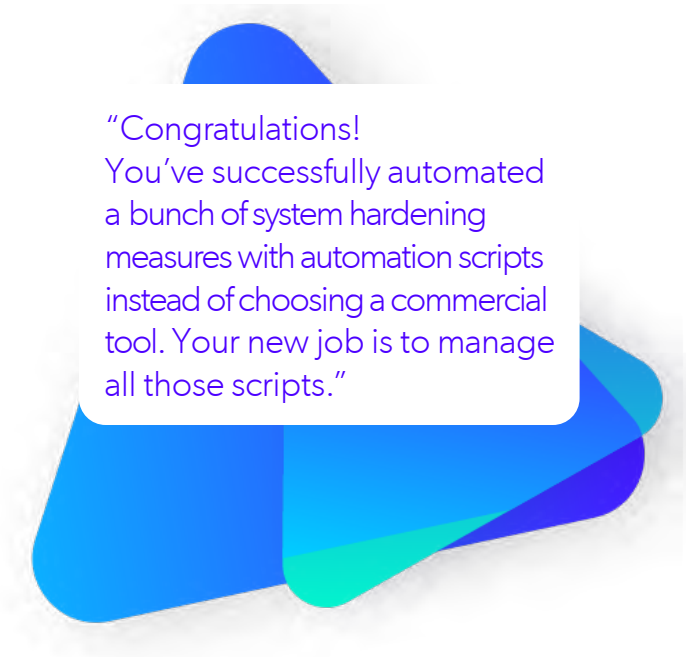
Writing your compliance policies as Puppet code — including system hardening configurations — immediately creates a central source of configuration management. Not only does that mean you've got an automated change record (when paired with version control and your CI/CD pipelines), but it also means you can control policies for all of the system resources Puppet manages, from servers to VMs and containers, with one platform.

Standardizing configurations means you can build predictability into your provisioning process, no matter what OS or deployment model you're on. You can provision Windows servers pre-configured with middleware, Linux VMs with the right network configurations, and containers with the right base images and rest easy knowing that those new machines have all the right security configurations applied.

Additionally, each hosting environment you use probably comes with its own configuration management tools (like AWS Systems Manager, Google Cloud Config Connector, and Azure Policy). The benefit of using a central configuration enforcement solution like Puppet is that you're not locked into vendor-specific tooling (or SMEs for each platform) to set, customize, and update configurations.

## A Solution That's Ready to Go vs. Months of Error-Prone Scripting

To buy or to build? That is the question! Building a system hardening strategy without automation is definitely possible (at least, up to a certain scale), but it requires a lot of up-front planning and effort — not to mention ongoing configuration, drift remediation, auditing, and risk management.




There are three key reasons why business leaders choose automation and configuration management platforms to maintain system hardening, rather than building and maintaining it all in-house:

1. **Scripts don't scale.** A thorough system hardening strategy means using, maintaining, updating, and replacing the tools we mentioned in the checklist portion earlier. Using more tools and creating a more sophisticated policy means more automation scripts to maintain that policy. And as you can guess, more scripts mean more problems: Automation scripts are inherently limited in scalability because they're often created for bespoke use cases, environments, or systems.
2. **You don't know when unauthorized changes throw you out of compliance.** Policy as code gives you centralized configuration management, including visibility into what configurations changed, when and by whom, and even how your compliance posture changed as a result. In contrast, you might only know something's wrong when your homebrew automation scripts stop working. By then, it could mean weeks of troubleshooting, tickets, and paperwork just to get things back in order.
3. **Tech debt accumulates when the script writers leave.** Turnover is already a huge issue in IT, and the cybersecurity skill gap gets wider with each generation. Building an automation patchwork that leverages the skills, expertise, and workflows of today's team can quickly turn it into a liability when you need to staff up or backfill.

In the end, most organizations find that they haven't saved very much time or money by building their own system hardening automation toolkit.

Compare that to **Puppet Enterprise Advanced**, a proven, agent-based automation and configuration management platform that enables policy as code for continuous compliance:

- **Desired state management means you can describe your desired configuration state** and Puppet will take the steps to get there.
- **Standardized Puppet manifests (called modules) automate common system hardening configurations** across Linux, Windows, and macOS.
- **The built-in CIS-CAT® Pro Assessor continuously monitors your configurations** for alignment with CIS Benchmarks, industry standard instructions for cybersecurity configurations.
- **Your configuration policies are written in Puppet DSL,** a proprietary language based in Ruby, as infrastructure as code (IaC) that can be audited and updated by your Puppet champion.
- **Built-in RBAC secures Puppet (and your infrastructure code) against unwanted or unauthorized changes** without limiting usability.

## Demo Puppet for System Hardening

Puppet Enterprise Advanced has capabilities and integrations that today's organizations need to manage complex IT, including system hardening measures. See them in action, from scanning and assessment to enforcement and remediation, by requesting a demo of Puppet.

**Demo Puppet ▶**