

**<Company Name>**

**Cyber Security Team Blue Team XX**

**Recipient**:

<Recipient Name>

<Orginization> , <Recipient Location>

**Date**: March X, 2025

**Subject:** Response to Request <Insert Request Title/Number>

Dear <Recipient/Team>,

Thank you for reaching out to our team. We have received the recent request # made to our team. We were asked to review our current patching policy as well as define a new policy that uses common industry and government standards. **We were also tasked with deploying this policy to the company infrastructure**. Below, you will see our report and recommended password policies.

Thank you,

Blue Team XX

Executive Summary

Our team observed that the procedures for upgrading our company infrastructure are undocumented and ill-defined. Although adequate for personal systems and use, this does not conform to the standards described by the National Institute of Standards and Technology (NIST), which are commonly used as the industry standard. As described in NIST Special Publication 800-40r4 (Guide to Enterprise Patch Management Planning): “Software used for computing technologies must be maintained because there are many in the world who continuously search for and exploit flaws in software. Software maintenance includes patching, which is the act of applying a change to installed software – such as firmware, operating systems, or applications – that corrects security or functionality problems or adds new capabilities.”

It is vital that we have a robust patching policy which is able to quickly prevent potential vulnerabilities in our systems while minimizing downtime – to prevent those who continuously search for and exploit flaws in software as the NIST standard describes. In order to conform to the NIST standard, we should follow the Software Vulnerability Management Life Cycle, described below.

After determining patching best practices, our team was able to deploy this policy on **X systems** on the company infrastructure while maintaining service availability to our customers throughout the process. In addition, our team has audited these systems and created a software and asset inventory.

Patching Policy

Our patching policy is based on the [NIST SP 800-63B 2024](https://pages.nist.gov/800-63-4/sp800-63b.html) document.

Our patching policy ensures that all Linux and Windows systems are consistently updated to minimize vulnerabilities and maintain system security. To address previous gaps, we implemented automated tools for both operating system and third-party software updates. On Linux systems, we configured unattended-upgrades for automation, while Windows systems are now set to check for and install updates daily via Group Policy. We also ensured third-party software is included in the patching process using package management tools like winget and verified patch logs for compliance.

**A staging environment has been created to test patches before deploying them to production. For critical systems, we use blue-green deployment strategies to apply patches with zero downtime. This approach ensures seamless transitions between environments while maintaining service availability.** Automated audit tools, such as lynis for Linux and Windows Update logs, have been set up to verify installed patches and identify missing updates.

To ensure ongoing compliance, we scheduled regular patching intervals based on system priorities and criticality, following the guidelines in NIST SP 800-40r4. **Notifications of pending updates and patch deployment statuses are sent to administrators via email and a centralized dashboard, keeping all stakeholders informed.** This policy prioritizes security while minimizing service disruptions and maintaining system functionality.

Assets Affected

| Hostname | IP Address | Additional Notes |
| --- | --- | --- |
|  |  | **REMOVE IF NONE** |

Fulfilled Action Items

**1. Initial Analysis of Current Patch Management Practices:**

Our team analyzed the current patching process for Linux systems. We found that updates were applied manually and inconsistently, with no standard procedures in place. Below are screenshots of outdated system packages and logs showing gaps in patching.

<Screenshots of outdated packages and patch logs>

2. **Review of Windows Systems Update Practices:**

Our team analyzed the Windows systems' patching practices and found that automatic updates were disabled by default, leaving systems vulnerable to unpatched exploits. We enabled Windows Update and configured it to check for updates daily. Below are screenshots of the previous configurations and the applied changes.

*<Screenshots of Windows Update configuration before and after>*

3. **Verification of Third-Party Software Updates:**

Our team identified that third-party software (e.g., browsers, productivity tools) on both Linux and Windows systems was not being patched. We implemented automated patching mechanisms using tools like winget on Windows and unattended-upgrades on Linux to address this issue. Below are screenshots of configured automation tools and patch logs.

*<Screenshots of third-party patch automation configurations>*

4. **Research and Application of Industry Standards:**

Our team researched industry standards for patch management and referenced [NIST SP 800-40r4 "Guide to Enterprise Patch Management Planning" (2024)](https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-40r4.pdf). Following these guidelines, we implemented a structured patching policy that prioritizes critical patches, defines patch testing in staging environments, and ensures timely deployment. Below are screenshots of our revised policies and patching schedules.

*<Screenshots of patching policy documents and scheduling configurations>*

6. **Automated Patch Auditing and Reporting:**

Our team configured automated auditing tools, such as lynis for Linux systems and built-in Windows Update logs, to monitor installed patches and identify missing updates. We created automated reports to provide insights into compliance and potential gaps. Below are screenshots of audit logs and sample compliance reports.

*<Screenshots of audit logs and generated reports>*

7. **Testing and Validation of Patched Systems:**

Our team conducted rigorous testing of patched systems in the staging environment. Tests confirmed that system functionality was unaffected, vulnerabilities were mitigated, and compatibility issues were resolved. Below are screenshots of test results, logs of successful updates, and a sample rollback procedure for failed updates.

*<Screenshots of test results, successful updates, and rollback logs>*

\*Delete below before turning in

If inject requires implementation, make sure you DO everything that was enumerated above. It is probably a good idea to install lynis on all the linux machines and ideally hook them up to SOC