Top 10 Patch Management Best Practices

Software development is a continuous process, whether it involves creating a Windows Server or a simple text editor. Vendors release functional and security updates and patches on a daily basis to fix bugs, add new features, and address compatibility issues and security vulnerabilities. While it may be easy to keep one workstation up to date, it can be challenging for organizations with multiple servers, databases, network devices, and other applications and services to manage the stream of updates. Even experienced specialists can get overwhelmed by the volume of updates, which highlights the need for a patch management process.

Patch management is the process of identifying, testing, and installing software updates and security patches on computers, servers, and other devices to address vulnerabilities, improve security, and enhance functionality. It is an essential aspect of IT infrastructure management because it helps ensure that the devices and systems used are up-to-date, secure, and functioning optimally.

Patch management is important for several reasons:

- **Security** - Patches can help fix vulnerabilities that hackers or malware could exploit, protecting sensitive data from unauthorized access or compromise.
- **Performance** - Updates and patches can improve the performance of devices and systems, making them run more efficiently and effectively.
- **Compliance** - Some regulatory requirements mandate the installation of certain patches and updates. Failing to comply can result in penalties or fines.
- **Productivity** - Keeping devices and systems up to date can minimize downtime and disruption, improving productivity and reducing the costs associated with repairing or replacing outdated or malfunctioning equipment.

Overall, patch management is a critical aspect of IT infrastructure management that is often overlooked. It helps ensure the security, performance, and compliance of devices and systems. This post highlights the key points of the patch management process, which can be tailored to the size and needs of your enterprise to simplify patching, avoid problems, and minimize downtime.

What Should Patch Management Process Consist of?

In order to have an effective patch implementation process, the following best practices should ideally be included:

1. **Establish patch management policies:** Clearly defined procedures will ensure that patches are applied consistently and in a timely manner.
2. **Take an inventory and consolidate your systems:** Knowing which systems and applications you have and where they are located can simplify the patch management process.
3. **Categorize and assign risk levels:** Assessing the risk level of each asset can help prioritize patches and ensure that the most critical systems are patched first.
4. **Monitor vendor patch and vulnerability newsletters:** Staying informed about potential risks and new patches can help anticipate software functionality changes and ensure that patches are deployed efficiently.
5. **Automate patch management:** Automation can streamline the process of identifying, acquiring, and deploying patches and ensure that patches are applied consistently.
6. **Expect patch exceptions**: There may be situations where it is not possible, preferable, or advisable to apply a particular patch, and organizations should be prepared for such scenarios.
7. **Test patches first:** Testing software before deployment to production systems is a best practice. It can identify potential issues and ensure that patches do not cause more problems than they solve.
8. **Create a backup:** Creating a backup before patching allows organizations to revert their systems to a known good state in case of issues caused by updates.
9. **Apply patches ASAP:** Applying patches as soon as possible can reduce the risk of vulnerabilities being exploited and keep assets up to date.
10. **Document patch implementation:** Documenting the patching process enables organizations to track the state of their systems and applications and identify any potential issues.

Implementing these best practices will simplify patch management and reduce the likelihood of security breaches and downtime.

Patch Management Best Practices

Although the list above may seem lengthy, reviewing each item reveals that the patch management process is actually simple and straightforward.

01. Create patch management policies

To implement a patch management process effectively, it is important to create a patch management policy that outlines the necessary steps and procedures. By doing so, organizations can ensure that their patching process is consistent, efficient, and effective.

For example, consider the 2017 WannaCry ransomware attack that affected hundreds of thousands of computers in over 150 countries. The attack exploited a vulnerability in Microsoft Windows operating systems for which a patch had been available for several months before the attack occurred. However, many organizations failed to apply the patch, leaving their systems vulnerable to the attack.

One such organization was the UK's National Health Service (NHS), which was particularly hard hit by the attack. The NHS was forced to cancel almost 20,000 appointments, including surgeries and diagnostic tests, as a result of the attack. The NHS's failure to apply the patch was due in part to a lack of a comprehensive patch management policy.

To prevent such attacks, a patch management policy should include the following elements:

- **Scope** - defines the types of devices and systems that the patch management process will cover, as well as the types of updates that will be installed.
- **Frequency** - outlines how often patches and updates will be checked and installed, including the required timeframes.
- **Testing** - is a process for testing patches and updates before installation to ensure that they do not cause any issues or conflicts.
- **Approval** - specifies who is responsible for approving patches and the criteria to be met before installation.
- **Communication** - describes how updates and changes will be communicated to relevant stakeholders, including IT staff, users, and management.

02. Inventory and consolidate your systems

One of the initial steps in implementing an effective patch management process is to conduct an inventory and consolidation of your assets. For example, a large financial institution may have hundreds or even thousands of devices across multiple locations, including desktops, laptops, servers, and mobile devices. It can be challenging to keep track of all these assets, their configurations, and their patch status.

To address this challenge, the financial institution conducts an asset inventory and consolidation exercise, which involves identifying all devices and systems used in the organization and creating an up-to-date list of their configurations, software, and patches. This information is crucial in ensuring that all assets are included in the patch management program.

Asset inventory and consolidation offer several benefits, including:

- **Efficiency**: By having a comprehensive list of assets, you can easily identify which systems need to be patched and ensure that patches are consistently installed across all devices.
- **Security**: By tracking which devices and systems are running older or vulnerable software, you can prioritize patching to minimize the risk of a security breach.
- **Cost savings**: By identifying any unnecessary or redundant systems that can be decommissioned, you can reduce costs and optimize your IT infrastructure.

03. Categorize and assign risk levels

In order to establish an efficient patch management process, the next step is to categorize and assign risk levels to available updates. This allows for prioritization of patches based on the potential risk associated with the vulnerabilities they address, ensuring that critical vulnerabilities are addressed first. For example, let's say a security team at a financial services company has identified a critical vulnerability in their trading platform that could allow hackers to manipulate financial transactions. This vulnerability is considered high-risk, and therefore should be given the highest priority for patching.

Several aspects should be considered during this step, including:

- **Criticality of the vulnerability**: Vulnerabilities that could result in data loss, downtime, or system compromise may be considered more critical and given higher priority.
- **Probability of vulnerability exploitation**: Patches that address vulnerabilities actively being exploited by hackers should be given higher priority.
- **Impact of vulnerability**: Vulnerabilities that could affect a large number of systems or have significant consequences may be given higher priority.

04. Monitor vendor patch and vulnerability announcements

To effectively manage patches, keeping track of vendor announcements is crucial to stay informed about new patches. This can be achieved by subscribing to vendor newsletters or checking Vulnerability Digests regularly. By monitoring patch announcements, organizations can:

- **Stay up-to-date** with the latest information and ensure that they are aware of new patches as soon as they are released, allowing them to take prompt action.
- **Address vulnerabilities** by installing updates as soon as they are available, thereby reducing the risk of data breaches.
- **Improve functionality**, as new patches and updates may include new features or improvements that can enhance the performance and usability of applications and devices.

05. Automate patch management

Automating the patch management process can be achieved using various tools and solutions. It is essential to choose the right tool or solution that suits your specific needs and ensure that it is appropriately configured and maintained.

For instance, patch management software like Action1 can help automate the process of identifying, acquiring, and deploying patches. These tools allow you to specify which systems and applications need to be patched and will automatically retrieve and deploy the necessary patches.

06. Expect patch exceptions

In some cases, it may not be feasible or advisable to install a specific patch, resulting in what is known as a patch exception. Patch exceptions can be caused by various factors, including:

- **Compatibility problems** - a patch may not be compatible with a specific system or application, either because it is not designed for that particular version or because it conflicts with other software or configurations. Customized systems or applications may also be incompatible with the patch.
- **Dependency problems** - a patch may depend on other patches being installed first, which may not be feasible on a particular target system.
- **Testing and evaluation** - before installation, the patch may require testing or additional assessment to assess its impact.

To avoid patch exceptions, it is critical to have a clear understanding of your systems and applications, as well as any customizations in your environment. It is also necessary to document the reason for the exception and to have a plan in place to address it by finding a workaround or alternative solution, deploying a different patch, or upgrading to a newer version of the system or application.

For example, suppose an organization runs a custom-built database management system that is not compatible with a critical security patch. In that case, the organization may need to consider other security measures, such as implementing network-level security controls or upgrading to a newer version of the database management system. By having a plan in place to address patch exceptions, organizations can ensure that their systems remain secure and up-to-date.

07. Test patches first

Testing patches before deploying them to production systems is a crucial step in the patch management process. By testing patches, you can ensure that they don't cause any issues when applied and can help identify potential problems, such as downtime or patch exceptions before they impact your operations. For example, let's say that a software development company recently identified a critical vulnerability in one of its key applications that could result in a data breach. After obtaining a patch from the vendor, the company's IT team performs testing to verify that the patch can be safely deployed without causing any issues to the application's functionality or performance.

To ensure effective testing, consider the following key points:

- **Scope**: Identify which assets will be affected by the patch and determine the testing scope. This may involve individual testing or a series of tests on a group of setups.
- **Test environment**: The test environment should closely mirror the production environment.
- **Testing process**: Observe the results, changes, and impacts of patching, which may include verifying that the patch has been correctly applied, running simulations of specific conditions, and analyzing system logs.
- **Evaluation and documentation**: Evaluate the results of patch testing to determine whether the patch can be safely deployed and document testing results, including any issues or potential impact identified. Documentation can be used to inform stakeholders or for future decisions.

08. Create backups

Example: A company experienced a critical issue after applying a patch that resulted in the loss of all their data. They were not able to recover the data and had to start from scratch, which led to significant downtime and financial losses. Had they backed up their system before applying the patch, they could have easily restored it to a previously known good state and avoided the issue.

Backing up your system is crucial to restoring it to a previously known good state in the event that a patch causes any issues. The backup workflow typically consists of the following milestones:

- **Identify which systems and applications need to be backed up**. This may include critical systems, systems with customized configurations, or data that would be difficult to restore.
- **Choose a backup method** that suits your setup and business needs. You can use a software solution, create a snapshot via native means, or manually copy important data and configurations. You can either make full or incremental backups.
- **Create the backup**.
- **Test the backup** by restoring it to a test system or application and verifying that all data and configurations are intact.
- **Store the backup** in a safe and secure location, such as a separate device, server, or cloud storage.

09. Apply patches ASAP

Applying patches promptly is a crucial patch management best practice because vulnerabilities or other issues are typically known by the time the patch is released, which can be exploited by attackers or cause system problems. However, in addition to recommended patch testing and evaluation, several factors should be considered:

- **Dependencies and compatibility**: Some patches may rely on other patches or may not be compatible with certain systems or applications.
- **Scheduling and downtime**: Updating may necessitate downtime for systems and business applications. To minimize the impact, it is necessary to schedule it appropriately.
- **Communication**: It is critical to communicate with relevant stakeholders, including users and IT staff. Advance notice of any required downtime or disruptions should be provided, and the advantages of current updates should be explained.

10. Document patch implementation

Effective patch management requires thorough documentation of patch details and deployment. Keeping track of this information allows you to identify potential issues, maintain up-to-date systems and software, and establish a baseline for future patching efforts. Here are some key pieces of information that should be documented:

- **Patch details**: include patch name, version, release date, and any relevant identification numbers. This information can be used to identify the patch and determine whether it has been applied correctly.
- **Systems and software affected**: document the specific systems and applications that the patch is intended for, and ensure that this information is kept up-to-date as changes are made to your environment.

- **Patch testing and evaluation results**: document the results of any patch testing and evaluation that has been conducted, including any issues or impacts identified during the testing process.
- **Patch deployment details**: document the date and time the patch was deployed, who applied it, and any issues during deployment.

Conclusion

In conclusion, implementing effective patch management practices is essential for maintaining the security and functionality of your IT environment. By following the top 10 best practices outlined in this guide, you can establish a robust patch management process that minimizes the risk of security breaches, system downtime, and data loss. Through real-world examples, we have demonstrated how organizations can categorize and prioritize patches, automate the patch management process, handle patch exceptions, test patches before deployment, and backup systems to ensure successful recovery. Remember that patch management is an ongoing process that requires continual attention, evaluation, and adjustment to stay effective. By implementing these best practices, you can ensure that your organization is well-prepared to handle any security threats and maintain the integrity of your IT systems.

Improving Your Patch Management with Action1

Instead of relying on outdated methods, consider utilizing a more efficient cloud-based patch management solution.

Action1 offers a range of benefits, including:

- Full automation and control of the entire patch management deployment cycle, from identifying missing updates to managing and distributing updates for Microsoft solutions and third-party applications, as well as compliance reporting.
- The ability to obtain a comprehensive list of patches, fixes, and pending OS updates across all of your computers.
- Customizable patch policies that can be configured to meet your specific needs, allowing you to push patches in accordance with your policies.
- Automatic monitoring of newly released patches, ensuring that your systems remain up-to-date.
- The ability to deploy software and run custom scripts no matter where your endpoints are located.