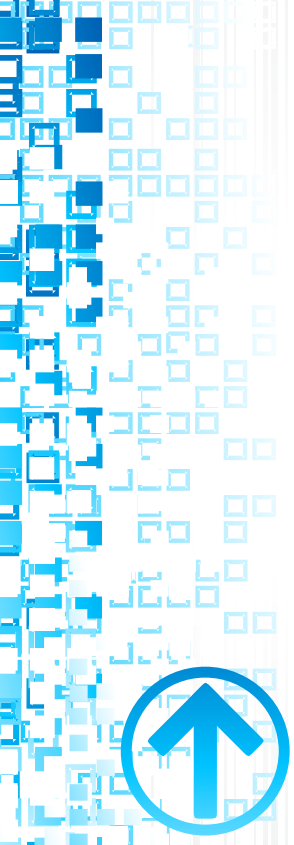
Taking a Proactive Approach to Linux Server Patch Management

Linux server patching

In years past, Linux server patch management was often thought of in terms of “we don’t patch our servers unless there is a reason to upgrade the version for application compatibility.” This philosophy is no longer appropriate today because of the downtime that can result from malicious code targeting known vulnerabilities on unpatched systems and the concerns around governance and regulatory compliance standards such as HIPAA (Health Insurance Portability and Accountability Act) and SOX (Sarbanes-Oxley Act). Patch management has now become an important buzzword in corporate IT organizations and business offices.



Taking a Proactive Approach to Linux Server Patch Management

TABLE OF CONTENTS

Patches are updates that incorporate changes in source code. They can be applied to the Linux kernel or to applications and other systems code running on a Linux server. Patches update security, incorporate new features, and fix coding errors to address such issues as application and Linux system performance and employee productivity.

The concept of patch management for Linux servers is simple, but developing processes for making it work well are not always simple. Patch management is basically the process of acquiring, testing and installing multiple code changes (patches) to systems software and applications. It includes maintaining current knowledge of available patches, determining which patches are appropriate, ensuring that patches are installed properly, testing patches in the target code after installation, and documenting all patches and the configuration(s) in which each patch was deployed. Patch management can be either manual or automated, or it can be some combination of both, which is the more common practice today.

Why patch Linux servers?

The most important reason to patch your Linux server is to maintain a secure environment for your servers' applications. Applying security patches updates your servers and plugs potential security holes left by outdated software or poorly written applications. The best way to keep your Linux servers secure is to keep up with the security alerts.

Maintaining an up-to-date working environment is another reason to patch your Linux servers. By updating your software, you have the opportunity to use the new features in software packages. However, maintaining Linux servers via patching will sometimes present issues around dependencies. For example, if a mistake is made during the patching of a Linux operating system, software dependency conflicts that prevent the updating of other software, such as business applications, can result.

During development, coding errors arise. When these errors are subsequently found, they are patched with bug fixes. These fixes are often required in the first year or so of a new release of a Linux operating system or application code.

To qualify for support from vendors, a supportable (from the vendor's point of view) version of Linux should be maintained. Running older, unpatched versions of Linux can be expensive to support and may be excluded from vendor support.

Linux kernel patches are typically viewed as different from other patches. They should be separated from the patching of other software running on Linux servers. Kernel patching often requires a restart of the system, whereas patching other software running on the Linux server may not require a reboot of the server.

Ways to patch a Linux server environment

While most IT organizations would like to have a fully automated process for patching Linux servers, this is not often the case. Instead, they may use some combination of manual patching, patching tools that come with Linux distributions, such as SUSE's YaST, and third-party patching tools that download Linux packages from the vendor and then perform the patching.

Manual patching is acceptable if you have the expertise to do it and you have only a few servers to maintain. The problem with manual patching is that you may have dependency issues that turn a simple patching effort into a long and complex exercise. As with manual patching, patching tools that ship with the distribution require technical expertise.

Third-party tools generally download Linux packages from the vendor, store the updates in a central repository and then install the patches to Linux servers. The goal of IT organizations with tens or hundreds of Linux servers is to be able to automatically install patches. Today, there are tools to do this. They save time and staff resources, reduce errors and allow the creation of automated processes for handling Linux server patch management.

IT organizations must take a proactive approach to Linux patch management. This white paper describes the importance of patch management and the challenges, and highlights the importance of automating patch management and following best practices. In addition, it examines a new topic that is quickly increasing in importance: patch management in cloud environments.

Linux server patching should be part of the corporate change management process

Patch management is viewed in many corporations as part of change management. All Linux system modifications, including patches, are performed and tracked through the change management system. Change management is the process of keeping track of all of the details of a system, such as which Linux release is running on each computer and which patches have been applied.

What to patch and when to patch

A good Linux server patch management strategy involves determining what to patch and when to install patches. If a patch resolves a security vulnerability or improves the performance of a Linux server but renders a piece of corporate business useless, a company can suffer the same net results of being hacked or running a slow application.

A process that answers the following questions can help determine what should be patched as soon as possible and what can be delayed:

- Will applying or not applying a patch impact critical business systems?
- Are there mitigations in place that reduce the threat?
- Will a large number of systems and/or users be affected if an attack exploits a vulnerability that hasn't been addressed by a patch?
- What would be the risk if the Linux system were left unpatched? Will performance suffer? Will the system be more vulnerable to compromise?

By providing answers to these questions, you can begin establishing priorities for patching.

Challenges for Linux server patch management

Linux server patch management presents several challenges, including handling the ever-growing number of security threats, managing the constant stream of patches and dealing with the growing number of physical and virtual servers to patch. Another big hurdle is just getting the organization to focus on patching.

One of the biggest patch management challenges is patch automation, which can provide huge benefits in terms of creating a cohesive patch management process and deploying Linux server patches more quickly and in an organized manner. The benefits of automating patch are several:

- **Security** – This is the most obvious reason why companies should seek an automated patch management solution. Applying Linux server security patches quickly and with the correct priorities

reduces problems such as data loss and inability to meet legal requirements.

- **Corporate productivity** – Failure to deploy patches quickly can affect corporate productivity. For example, performance patches that fix low-performing locking mechanisms in a Linux operating system or patches that fix Linux system crashes are critical non-security patches because they can affect employee productivity.
- **Reduce time-consuming manual patching by IT** – When the number of Linux servers in an organization grows to more than 40 or 50, the amount of time required to perform manual patching is so excessive that IT folks are often relegated to fixing only those patches with urgent and high priorities. You need to ascertain how many people and how many working hours are devoted to manually patching and determine how much more efficient the IT organization could be by automating at least some parts of the Linux server patch management process.
- **Regulatory compliance** – Various laws and regulations impose security best practices on companies. Having Linux systems fully patched is one of the most important security rules. Failure to comply can result in legal and financial penalties and lost business opportunities.

Patch management best practices

There are many best practices for Linux server patching, but the following are among the most important:

- **Inventory what you have, so you know what you need to protect and patch** – An inventory of Linux server software versions, releases and patch levels in production software enables automatic matching of incoming patches against the software inventory.
- **Automate as much of your Linux server patch management as possible** – There are a number of patch management tools for Linux servers available. Automated patch management shortens the time between when a patch is received and when it is tested and then deployed, freeing IT staff to focus more attention on other responsibilities.
- **Review associated patch documentation** – Review the readme information and other related documentation, including prerequisites, issues, functionality changes and alternative workarounds that will be important in the evaluation of a Linux server patch. Do not apply a patch until you fully understand whether it can damage (by applying it or not applying it) your business.
- **Assign a priority to each Linux server patch** – Having a patch priority mechanism in place allows you to determine which Linux server patches are the most important and should be deployed first (urgent, critical, medium, low).
- **Fully test Linux server patches before deployment in production code** – After a Linux server patch is authorized, it should be tested in a lab-like environment using Linux systems that simulate the current Linux production environment. Lacking a lab-like environment, deploy the patch in a small, controlled group of production machines, with rollback procedures in case of problems. Do not certify a patch (for deployment) without testing it in a production-like Linux environment.
- **Notify end users and administrators that a Linux server patch has been deployed** – After a patch has been certified, notify end users and administrators about the deployment of the patch and what steps they should follow for reporting problems.

Patch management in cloud environments

Patch management in Linux-based cloud environments is often a function of consistency and automation. If you have a single private cloud or public cloud, or a hybrid cloud built by a single cloud provider with the same infrastructure, then consistency across the cloud(s) may exist. If so, patch management in Linux-based cloud environments is somewhat practical.

There are basically two ways to handle Linux server patch management in cloud environments: in-place update or in-place swap. The in-place update approach is the same way Linux servers are patched in non-cloud environments – take an existing Linux server and update the patch level of the Linux operating system, applications, etc. You can have a replicated environment for testing the patches before deploying them, or you can patch one virtual server in the production environment and test it to determine if the patch works as expected. The in-place update approach works best when leveraging patch automation to drive patches in scale. As in the non-cloud environment, the test environment must be an exact duplicate of your production environment.

The in-swap approach rebuilds all the Linux servers with the updated patch level, deploys the applications to those updated Linux servers and cuts the traffic over from the old (non-patched) machines to the new (fully patched) machines. The in-place swap allows you to keep the old environment around for a while in case you find a bug or some type of error in the patching process.

Summary

While patch management may be viewed by some companies as "just something that we have to do when we have time," it is clear that it is an important issue. Managing patches for all of your Linux servers, even in a small company, is fairly complicated and time consuming, especially if performed manually. The tips provided in the best-practices section of this paper can help guide you toward the development of a good patch management process for Linux servers and prevent the consequences associated with failing to deploy patches in a timely manner.