Essential Guide to Prioritized Patching

WHITE PAPER

Balbix®

Essential Guide to Prioritized Patching

What is patching prioritization and why it matters

With software running everywhere and on pretty much every imaginable device, patching security vulnerabilities has become a major challenge. In a simpler world, enterprise security teams would just identify the vulnerabilities and apply available patches to them. But where do you begin when you have tens of thousands of vulnerabilities across hundreds of thousands of assets that potentially need to be patched? How do you quantify your exposure, assess business risk, and prioritize patching so that you fix your most critical vulnerabilities first?

Imagine that you're an IT professional responsible for protecting your enterprise from cybersecurity threats. You're expected to know the inventory of assets in your environment, the software running on each of them, the known vulnerabilities, and what patches are available to fix those vulnerabilities. Once that's done, you need to identify your highest risk exposures so you can patch software applications according to their criticality to the business.

Now let's say your environment includes 50,000 assets with 100 open known vulnerabilities (CVEs) on each asset, on average. This translates to 5,000,000 vulnerabilities that potentially need to be patched. Clearly, in modern resource-constrained security teams, patching everything has become impossible. We need a prioritization strategy that achieves maximum reduction in breach risk for minimum patching effort. The business criticality of assets can range from mission-critical to insignificant and everything in-between. The key question is – where do you begin?

Patching 101

Software applications ranging from critical infrastructure (power grids, financial systems), to operating systems (iOS, Microsoft Windows), to consumer applications (personal finance, healthcare, payment systems) rule our lives. And, because software development is a human endeavor, it is inherently prone to error. Vulnerabilities that get introduced in software development can be exploited by malicious actors to bring down critical assets, steal sensitive data, and damage a company's brand. To systematically reduce the risk of such breaches, software vendors or third parties periodically release *patches* specific to vulnerabilities that, when applied, render the software robust against known exploits. The practice of remediating known vulnerabilities by timely discovery and patching has come to be known as vulnerability management.

The lifecycle of vulnerability exploitation

From the time of introduction, the lifecycle of a single vulnerability goes through multiple stages involving several key stakeholders who each have different motivations and roles. From the point of view of vulnerability management, it is important to note that each stage can introduce data quality issues (coverage and precision), which in turn affect remediation strategy and effort. Here we discuss the lifecycle and stakeholders with a particular focus on data quality at each stage.

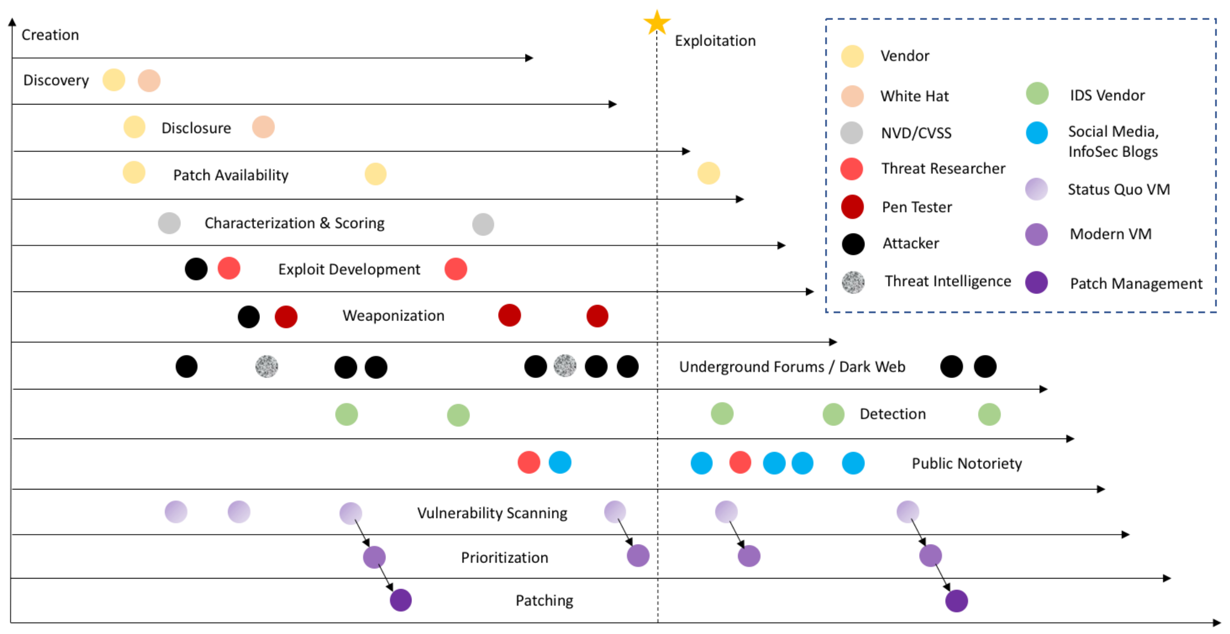

Figure 1: The Lifecycle of a Vulnerability

Discovery and disclosure

Vulnerabilities are introduced by software vendors as software gets developed. Through various means – verification, testing, and bug bounties, for example – vendors and white hat researchers discover vulnerabilities.

The MITRE Corporation and the US National Vulnerability Database (NVD) have formalized the coordinated public disclosure of vulnerabilities. Once disclosed, they are allocated a name by the CVE naming authority (CNA) and made available as Common Vulnerabilities and Exposures (CVEs).

To date, about 120,000 CVEs have been disclosed and made available through the NVD. This accounts for the majority of known vulnerabilities, but not all vulnerabilities are available through the NVD alone. Alternate sources are maintained by major software developers such as Microsoft (MS Bulletin), Oracle and Adobe, the Chinese government (CNNVD), Linux distributions, and private integrators.

Implications for Vulnerability Management: Lack of coordinated disclosure, high-quality curation, and centralized access to vulnerability definitions makes it hard for vulnerability management practitioners to scan their environment and identify which vulnerabilities are present.

Precision Issues	Status
Not a vulnerability	Very rare
Vendor releases patches only for less critical vulnerabilities	Very rare
Coverage Issues	**Status**
Zero days	Unknown
Uncoordinated disclosures	Rare
Missing patches for highly critical vulnerabilities	Very rare
Disclosures not curated by centralized databases or major vendors	Under 10%

Characterization and scoring

Soon after public disclosure, CVEs are characterized and scored using popular open industry standards: primarily the Common Vulnerability Scoring System (CVSS). The CVSS score factors in technical ease of exploitation, the access method, and the severity of impact to create a standardized score in the range of [0, 10]. The CVSS score can also be modified over time as exploit methods become known and mitigation techniques become available.

The CVSS score has become the most used metric to prioritize vulnerabilities for remediation. However, 2017 and 2018 have seen a massive uptick in the number of CVEs (over 30,000 of the total of 120,000 have been disclosed in the last two years) as well as the proportion of CVEs with high or critical scores (about 30% of all CVEs in the NVD are rated as high or critical: CVSS > 7).

Implications for Vulnerability Management: Since the late 2000s, CVSS scores have become the most popular axis for prioritizing which vulnerabilities to patch immediately and which ones to postpone. However, too many high or critical CVEs makes it difficult to tell the relative urgency of one from the other, leading to the need for further axes of prioritization.

Precision Issues	Status
Critical or high ratings for difficult to exploit vulnerabilities	Rare
Coverage Issues	**Status**
Vulnerabilities missing scores due to high volume or lack of resources	Rare but increasingly common

Patch availability

Disclosures can either be coordinated or not with the vendor. After a security patch is developed, vendors disclose the vulnerability and make the patch generally available. Occasionally, disclosure can also occur through a third party, and sometimes, without a patch. This is the typical scenario.

In a fraction of cases, for rare vendors or rare vulnerabilities, they are disclosed by a third-party researcher. In these cases, the vendor may take note after the disclosure and make the patch available soon after. In a smaller fraction of scenarios, there exists no known mitigation for the vulnerability even after disclosure. The problem of patch availability for vulnerabilities is a long-tail problem, with the majority of vulnerabilities addressed by a minority of vendors and products, and a minority of vulnerabilities associated with rare vendors, products, or versions.

Implications for Vulnerability Management: The long-tail nature of patch availability suggests that it requires nontrivial effort to discover and apply relevant patches for vulnerable software.

Precision Issues	Status
Ineffective patch	Very rare
Coverage Issues	**Status**
No known mitigation or patch	Under 10%

Public exploit development and weaponization

Once a vulnerability has been disclosed and characterized, exploiting it is still far from straightforward. Threat researchers try to develop exploit methodologies, often by reverse-engineering patches. Proof-of-concept (POC) exploit codes are then developed and shared in the public domain, curated by databases such as Exploit DB.

A central pillar of vulnerability management is the practice of penetration testing – ethically hacking into networks by discovering and exploiting vulnerabilities. As exploit code becomes available in the public domain, penetration testers apply exploit POC code to develop weaponized tools for real-world ethical hacking. Databases such as Metasploit curate reproducible, weaponized exploit kits which the penetration testing communities can share.

Implications for Vulnerability Management: Unfortunately, exploit development and weaponization also occurs in the attacker community. Attackers further have the advantage of public domain resources to develop their own exploits. Vulnerability management practitioners who want to further prioritize which vulnerabilities to fix will do well to prioritize based on the stage of exploit development (no exploit available to weaponize).

Precision Issues	Status
Not an exploit	Very rare
Coverage Issues	**Status**
Risk of zero days	Very rare

Attack markets and threat intelligence

For CVEs with unknown exploits, attackers can develop and trade exploits in underground markets enabled by the deep and dark web. Exploits form a continuum in this marketplace, all the way from highly priced exploits for zero days to chatter on forums regarding potential institutions or industry verticals to exploit.

The existence of attack markets creates the demand for threat intelligence. Specialized threat intelligence actors – commercial and state-sponsored entities – monitor attacker forums for specialized intelligence that can be used to prioritize vulnerabilities to fix based on their "fashionability" in the attacker community.

Implications for Vulnerability Management: Use threat intelligence in the right way to deepen your understanding of which vulnerabilities are most likely to be exploited in the current zeitgeist for your organization's size and industry vertical.

Coverage Issues	Status
Real markets and insights are traded on clandestine, inaccessible forums	Unknown

Detection of exploitation in the wild

The primary function of intrusion detection systems (IDS) is to detect and respond to exploits occurring in the wild. The exploitation of which vulnerabilities can and cannot be detected in the wild depends on the maturity of detection signatures. IDS developers use known exploits as a starting point to develop signatures that recognize attacks.

Implications for Vulnerability Management: Understanding which exploits have been detected in the wild are a further useful axis of prioritization. Successful exploitation in the wild feeds back to fuel attacker attention towards the vulnerability through public media and underground forums. Thus, vulnerabilities that have been previously exploited successfully are more likely to be targeted, resulting in a rich-club phenomenon in attacker behavior. However, this strategy suffers from a crucial caveat: detecting exploits in the wild depends on IDS signatures, which in turn suffer from an unknown false negative issue.

Precision Issues	Status
Imprecise IDS signatures	Unknown
Coverage Issues	**Status**
Exploitation of known vulnerabilities without detection signatures	Unknown
Zero days	Unknown

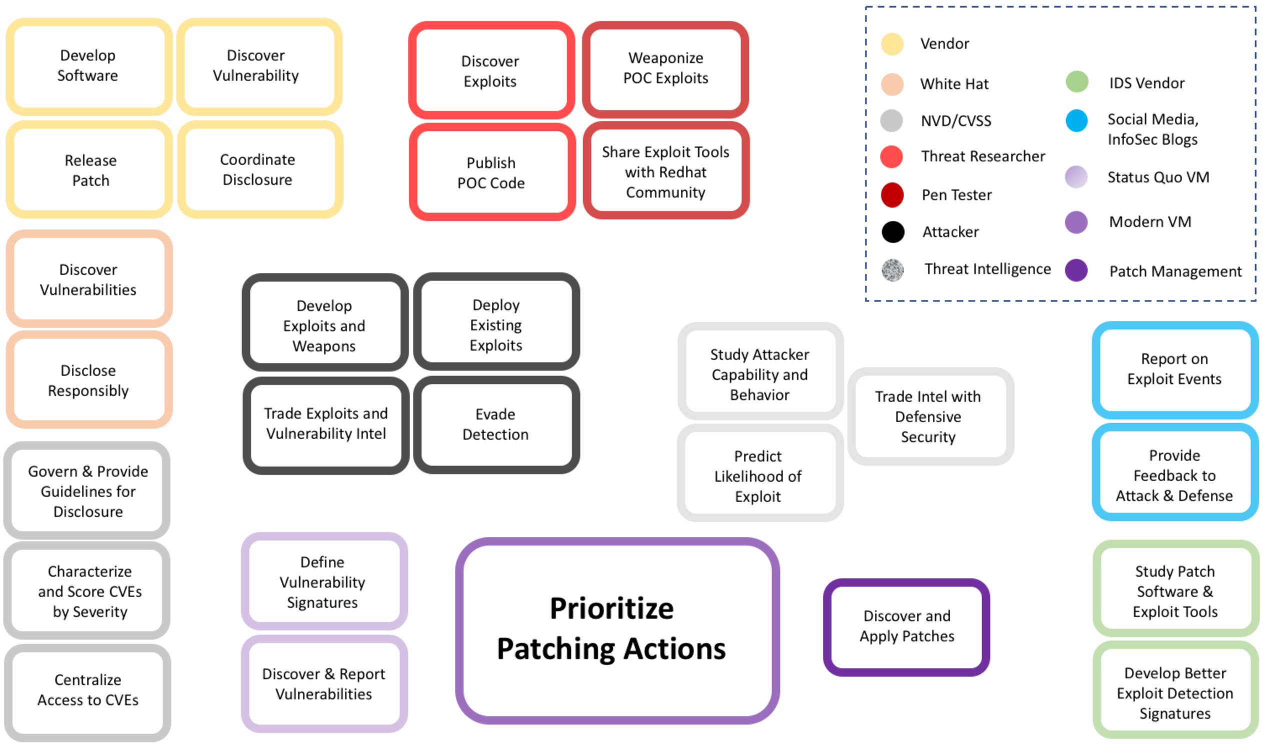
Develop Software
Discover Vulnerability
Release Patch
Coordinate Disclosure
Discover Exploits
Weaponize POC Exploits
Publish POC Code
Share Exploit Tools with Redhat Community
Vendor
White Hat
NVD/CVSS
Threat Researcher
Pen Tester
Attacker
Threat Intelligence
IDS Vendor
Social Media, InfoSec Blogs
Status Quo VM
Modern VM
Patch Management
Discover Vulnerabilities
Disclose Responsibly
Develop Exploits and Weapons
Deploy Existing Exploits
Trade Exploits and Vulnerability Intel
Evade Detection
Study Attacker Capability and Behavior
Trade Intel with Defensive Security
Predict Likelihood of Exploit
Report on Exploit Events
Provide Feedback to Attack & Defense
Govern & Provide Guidelines for Disclosure
Characterize and Score CVEs by Severity
Centralize Access to CVEs
Define Vulnerability Signatures
Discover & Report Vulnerabilities
Prioritize Patching Actions
Discover and Apply Patches
Study Patch Software & Exploit Tools
Develop Better Exploit Detection Signatures

Figure 2: Stakeholders and Roles

How "status quo" vulnerability management works

To really understand how vulnerability management works today, and where the gaps are, it is important to understand the lifecycle of vulnerability exploitation (see Fig 1).

To reduce risk from exploitation of unpatched systems, the security team must continuously identify the enterprise's open vulnerabilities, check for availability of patches, and then systematically roll them out to their assets. Rolling out patches is not a trivial task because software updates cause interruptions to the business. Also, each software update can potentially result in backward compatibility issues, demanding days or weeks of carefully testing newly patched systems in staging environments before they can replace high-availability systems in production.

In a previous era, when a few IT system administrators or security analysts had complete knowledge of their inventory, vulnerability management was a human scale problem. Using either freeware or commercial software, the security team would periodically scan the enterprise assets for known vulnerabilities, manually triage them to eliminate false positives, and then download and apply patches. Note that legacy vulnerability management software is good at only one thing: discovering open vulnerabilities in installed software. This capability, on its own, has become inadequate for a modern vulnerability management practice.

Patching everything is both harder and unnecessary

Today, we live in a new kind of world. Data centers, application servers, laptops, smartphones, networking and storage equipment, VoIPs, and IoTs are all commonplace. This long tail of internet-connected hardware is accompanied by a proliferation of software, and with it, a proliferation of vulnerabilities. This has had two consequences:

1. First, more assets and vulnerabilities mean a growing enterprise attack surface, moving vulnerability management beyond a human scale problem. There just aren't enough resources to patch everything.

2. Second, the outsized impact of vulnerabilities being exploited has also brought larger budgets and more attention towards research and disclosure. Indeed, of the ~120,000 known CVEs disclosed since the beginning of time, more than 30,000 have been reported in 2017 and 2018 alone.

Yet – and crucially – only a small fraction of all known CVEs is observed in a typical enterprise (10-30%), and an even smaller fraction (< 2%) is reported to be exploited in the wild based on public data available from IDS signatures (but see fig. 1 for caveats). This massive gap between all known CVEs, observed CVEs, and eventually exploited CVEs demonstrates that patching all assets is both unnecessary and a flawed vulnerability management strategy.

This leads us to prioritization. To be precise, without automated prioritization of which vulnerabilities to patch, the status quo vulnerability management practice is prone to priority inversion, i.e., precious effort is spent on the laborious task of discovering and applying patches which may or may not decrease breach risk.

In a recent *Ponemon study*, only 1 in 3 organizations is confident that it can avoid a data breach, and 63% say that they are unable to act on the large number of alerts and actions generated by their vulnerability management program.

Raising your patching strategy to a whole new level

There are three key steps involved in the patching workflow:

1. Vulnerability Scanning: Inventorying all open vulnerabilities on all assets
2. Prioritization Strategy: Prioritizing vulnerabilities to maximize business risk reduction
3. Patch Management: Identifying and applying available patches for prioritized vulnerabilities

3 Areas to Focus on to Prioritize Patching

1. Vulnerability scanning

Discovering what vulnerabilities affect your environment is the first step in vulnerability management. There are three key factors that affect the coverage and precision of your vulnerability scans:

Knowing what vulnerabilities to look for depends on the enumerated list of known vulnerabilities gathered from vulnerability databases (such as NVD, CNNVD, VulnDB).

Having a comprehensive inventory of assets depends on the quality of tools to continuously monitor your asset inventory. Evaluate whether your periodic inventory methods monitor managed and unmanaged assets across categories including servers, networking devices, storage assets, IoTs, OT, ICS systems, BYODs, and beyond.

Having a comprehensive inventory of software depends on the methods used to discover installed software across all of your assets. Agentless methods are less invasive but suffer from false positives. Endpoint agents do not scale to the long tail of asset classes (IoTs, OTs, etc.).

Precision Issues	Status
Uncoordinated distribution of patches across OS distributions and OEM vendors	Common
Imprecise mapping of software versions to vulnerabilities	Common
Coverage Issues	**Status**
Incomplete asset inventory	Very common
Incomplete software inventory	Common
Incomplete vulnerability coverage	Uncommon

2. Prioritization strategy

As discussed, prioritization is key to ensure that the effort of patching under scarce resources is expended towards the maximum risk reduction. Vulnerability management programs distinguish themselves by their quality of prioritization.

CVSS v3.0 prescribes a high-level framework to prioritize vulnerabilities (cite), broken down in what they call base factors — based purely on the technical factors of the vulnerability, temporal factors — based on how the likelihood of real-world exploitation can change over time, and environmental factors — based on attributes of the internal IT environment (see Fig. 3).

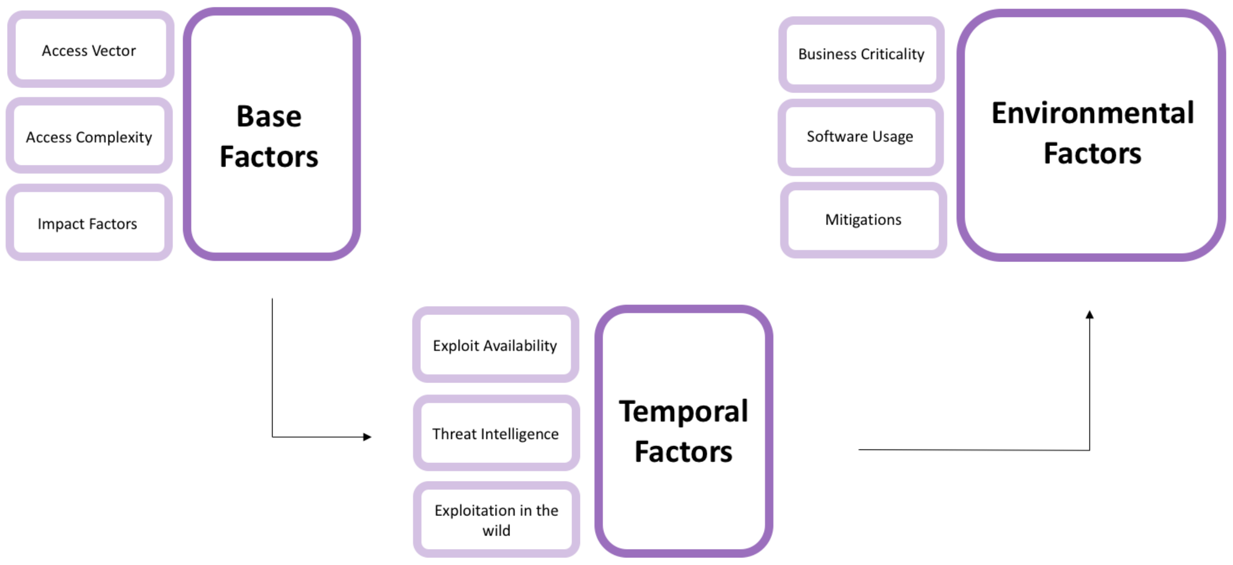

Figure 3. Mapping the CVSS v3.0 framework for vulnerability prioritization on to the vulnerability management practice.

Different vendors and vulnerability management practices adopt a variant of this basic framework, which we describe below as broken down into Threat-based and Risk-based strategies.

Threat-based strategy 1: Severity and Ease of Exploitation

Which vulnerabilities represent the greatest threats? One answer to this question is the CVSS score, a transparent and systematic method of quantifying the severity and technical ease of exploitation. The CVSS score is extremely popular, and increasingly mandated by private enterprises and state actors to define best practices around patching timelines (see the recent DHS directive).

Prioritization based on severity makes intuitive sense at first glance. Focusing on CVEs that are more severe should, in theory, limit the potential impact of a breach. However, this approach falls short because it does not take into account how likely an attacker is to exploit a given vulnerability. Furthermore, there are too many CVEs that are rated high or critical by CVSS, and therefore additional prioritization is often required, even among high or critical vulnerabilities identified by vulnerability scanners.

Threat-based strategy 2: Likelihood of Exploitation

As discussed in fig. 1 and elsewhere, the way a vulnerability gets created, disclosed, characterized, and eventually exploited is often informative regarding the likelihood that any given vulnerability will be used to attack your environment.

Modeling the “likelihood of exploit” is thus a useful axis of prioritization. Recent analysis suggests that two of the most important predictors of this factor are whether an exploit is available and whether it has previously been exploited in the wild. Additional factors to model come from threat intelligence regarding attacker behavior (chatter from underground forums and markets on the deep and dark web).

Advanced models of exploitation likelihood are a step in the right direction, but they fall short in two ways.

- They do not take into account the potential impact of each CVE on your organization.
- They do not tell you about exposures based on your use of particular software.

Risk-based strategy

Adopting a risk-based prioritization strategy can help overcome these shortcomings. The central pillar of a risk-based strategy is to make risk measurement comprehensive and real-time (as opposed to periodic) across asset inventory, software usage, and compensating controls.

Once you have real-time visibility into your assets and software use, a risk-based prioritization strategy affords three axes for prioritization.

Risk-based strategy 1: Business Criticality
First, ranking all of your assets by their business criticality helps prioritization. For instance, endpoint machines with administrative access to core servers can be prioritized ahead of contractor laptops or guest machines.

Risk-based strategy 2: Software Usage
Second, real-time visibility into the dynamics of software usage helps to prioritize the most used software. For instance, CVEs reported on Internet Explorer by vulnerability scanners can be safely deprioritized ahead of CVEs on software versions that are used every day on business-critical assets.

Risk-based strategy 3: Mitigations
Third, automatic discovery and effectiveness evaluation of compensating controls enable prioritization of assets with no mitigations in place. For instance, critical assets without endpoint mitigations may be prioritized ahead of hardened jump hosts that are difficult to access in the environment.

Lack of automation in ranking and prioritization based on risk factors outlined above can lead to priority inversion centered around issues of precision and coverage, some of which are stated below.

Precision Issues	Status
Focus on less business-critical assets	Very common
Focus on rarely used software	Unknown
Focus on assets with strong compensating controls	Unknown
Focus on vulnerabilities unlikely to be exploited	Common
Coverage Issues	**Status**
Lack of focus on patching privileged and business-critical assets	Very common
Lack of focus on most used and prevalent software	Unknown
Lack of focus on vulnerabilities most likely to be exploited	Common

3. Patch management

Once a clear patching strategy has been articulated and the vulnerabilities to be addressed have been prioritized, it is important to use effective tools to discover patches from vendors and automate patching at scale. Finding and using the right patch management tools can significantly reduce the effort involved in patching.

Key takeaways

- Prioritizing patching is paramount for good security posture because teams are chronically understaffed.
- Traditional vulnerability management is broken because it was built to discover vulnerabilities, not prioritize them.
- The problem of identifying whether a vulnerability exists on an asset is easy to get approximately right but hard to get exactly right. It is important to evaluate your vulnerability management scanner for its precision and coverage across asset discovery, software version enumeration, and vulnerability mapping.
- To reform vulnerability management , it is crucial to understand the lifecycle of vulnerability exploitation. Each step in this lifecycle is confronted by data quality issues. Making an upfront decision regarding which data quality risks to address will significantly clarify prioritization and remediation.
- Threat-based prioritization methods, an integral part of threat and vulnerability management (TVM), factor in the severity of a CVE and the likelihood of exploit, but lack context of your environment.
- Although TVM provides an illusion of focus, its quantification of risk is based on external factors. TVM has no reliable way to quantify real risk-reduction in your environment. This results in wasted effort tackling what may be popular threats in the zeitgeist without validating if they apply to you.
- Risk-based prioritization methods, such as risk-based vulnerability management, factor in the business criticality of assets, software usage, and the use and effectiveness of compensating controls. Done properly, risk-based vulnerability management achieves a massive reduction in effort for a quantifiable reduction in risk.
- Make sure that risk-based vulnerability management system is powered by a measurement of risk that comprehensively tracks your attack surface and is real-time, not episodic.
- Prioritize your patching activity by the business criticality of your assets, software usage, compensating controls, and global threats.
- Use patch management software to increase your efficiency discovering and applying patches.
- Finally, remember that patching of managed assets, however effective, addresses only a tiny fraction of your vast attack surface. Weak or default passwords, missing or poor encryption, lack of appropriate segmentation, misconfiguration, and insider threats are all equally important vectors, and must be measured, quantified, and addressed equally efficiently.

LEARN MORE

3031 Tisch Way, Ste 800
San Jose, CA 95128
866.936.3180
info@balbix.com
www.balbix.com