

3 **Guide to Enterprise Patch**
4 **Management Planning:**
5 *Preventive Maintenance for Technology*

6
7 Murugiah Souppaya
8 Karen Scarfone
9

10
11
12
13 This publication is available free of charge from:
14 <https://doi.org/10.6028/NIST.SP.800-40r4-draft>
15
16
17
18

19 **Draft NIST Special Publication 800-40**
20 **Revision 4**

21 **Guide to Enterprise Patch**
22 **Management Planning:**
23 *Preventive Maintenance for Technology*

24
25 Murugiah Souppaya
26 *Computer Security Division*
27 *Information Technology Laboratory*

28
29 Karen Scarfone
30 *Scarfone Cybersecurity*
31 *Clifton, VA*

32
33
34 This publication is available free of charge from:
35 <https://doi.org/10.6028/NIST.SP.800-40r4-draft>
36
37

38 November 2021
39



40
41
42
43 U.S. Department of Commerce
44 *Gina M. Raimondo, Secretary*

45
46 National Institute of Standards and Technology
47 *James K. Olthoff, Performing the Non-Exclusive Functions and Duties of the Under Secretary of Commerce*
48 *for Standards and Technology & Director, National Institute of Standards and Technology*

49

Authority

50 This publication has been developed by NIST in accordance with its statutory responsibilities under the
51 Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 *et seq.*, Public Law
52 (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including
53 minimum requirements for federal information systems, but such standards and guidelines shall not apply
54 to national security systems without the express approval of appropriate federal officials exercising policy
55 authority over such systems. This guideline is consistent with the requirements of the Office of Management
56 and Budget (OMB) Circular A-130.

57 Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and
58 binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these
59 guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce,
60 Director of the OMB, or any other federal official. This publication may be used by nongovernmental
61 organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would,
62 however, be appreciated by NIST.

63 National Institute of Standards and Technology Special Publication 800-40 Rev. 4
64 Natl. Inst. Stand. Technol. Spec. Publ. 800-40r4, 27 pages (November 2021)
65 CODEN: NSPUE2

66 This publication is available free of charge from:
67 <https://doi.org/10.6028/NIST.SP.800-40r4-draft>

68 Certain commercial entities, equipment, or materials may be identified in this document in order to describe an
69 experimental procedure or concept adequately. Such identification is not intended to imply recommendation or
70 endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best
71 available for the purpose.

72 There may be references in this publication to other publications currently under development by NIST in accordance
73 with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies,
74 may be used by federal agencies even before the completion of such companion publications. Thus, until each
75 publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For
76 planning and transition purposes, federal agencies may wish to closely follow the development of these new
77 publications by NIST.

78 Organizations are encouraged to review all draft publications during public comment periods and provide feedback to
79 NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at
80 <https://csrc.nist.gov/publications>.

81 **Public comment period: November 17, 2021 through January 10, 2022**

82 National Institute of Standards and Technology
83 Attn: Applied Cybersecurity Division, Information Technology Laboratory
84 100 Bureau Drive (Mail Stop 2000) Gaithersburg, MD 20899-2000
85 Email: cyberhygiene@nist.gov

86 All comments are subject to release under the Freedom of Information Act (FOIA).

87 **Reports on Computer Systems Technology**

88 The Information Technology Laboratory (ITL) at the National Institute of Standards and
89 Technology (NIST) promotes the U.S. economy and public welfare by providing technical
90 leadership for the Nation’s measurement and standards infrastructure. ITL develops tests, test
91 methods, reference data, proof of concept implementations, and technical analyses to advance
92 the development and productive use of information technology. ITL’s responsibilities include the
93 development of management, administrative, technical, and physical standards and guidelines for
94 the cost-effective security and privacy of other than national security-related information in
95 federal information systems. The Special Publication 800-series reports on ITL’s research,
96 guidelines, and outreach efforts in information system security, and its collaborative activities
97 with industry, government, and academic organizations.

98 **Abstract**

99 Enterprise patch management is the process of identifying, prioritizing, acquiring, installing, and
100 verifying the installation of patches, updates, and upgrades throughout an organization. Patching
101 is more important than ever because of the increasing reliance on technology, but there is often a
102 divide between business/mission owners and security/technology management about the value of
103 patching. This publication frames patching as a critical component of preventive maintenance for
104 computing technologies – a cost of doing business, and a necessary part of what organizations
105 need to do in order to achieve their missions. This publication also discusses common factors
106 that affect enterprise patch management and recommends creating an enterprise strategy to
107 simplify and operationalize patching while also improving reduction of risk. Preventive
108 maintenance through enterprise patch management helps prevent compromises, data breaches,
109 operational disruptions, and other adverse events.

110 **Keywords**

111 enterprise patch management; patch; risk management; update; upgrade; vulnerability
112 management.

113 **Acknowledgments**

114 The authors wish to thank their colleagues who reviewed the document and contributed to its
115 technical content, especially Mark Simos from Microsoft, and Jamie Brown, Vincent Gilcreest,
116 and Glen Pendley from Tenable.

117 **Audience**

118 The primary audience for this publication is chief information officers, chief information security
119 officers, cybersecurity directors and managers, and others who are responsible for managing
120 organizational risk related to the use of software. Others with an interest in the topic, including
121 business and mission owners, security engineers and architects, system administrators, and
122 security operations personnel, may find portions of the publication to be informative.

123 **Trademark Information**

124 All names are registered trademarks or trademarks of their respective companies.

125

Call for Patent Claims

126 This public review includes a call for information on essential patent claims (claims whose use
127 would be required for compliance with the guidance or requirements in this Information
128 Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be
129 directly stated in this ITL Publication or by reference to another publication. This call also
130 includes disclosure, where known, of the existence of pending U.S. or foreign patent applications
131 relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

132 ITL may require from the patent holder, or a party authorized to make assurances on its behalf,
133 in written or electronic form, either:

134 a) assurance in the form of a general disclaimer to the effect that such party does not hold
135 and does not currently intend holding any essential patent claim(s); or

136 b) assurance that a license to such essential patent claim(s) will be made available to
137 applicants desiring to utilize the license for the purpose of complying with the guidance
138 or requirements in this ITL draft publication either:

139 i. under reasonable terms and conditions that are demonstrably free of any unfair
140 discrimination; or

141 ii. without compensation and under reasonable terms and conditions that are
142 demonstrably free of any unfair discrimination.

143 Such assurance shall indicate that the patent holder (or third party authorized to make assurances
144 on its behalf) will include in any documents transferring ownership of patents subject to the
145 assurance, provisions sufficient to ensure that the commitments in the assurance are binding on
146 the transferee, and that the transferee will similarly include appropriate provisions in the event of
147 future transfers with the goal of binding each successor-in-interest.

148 The assurance shall also indicate that it is intended to be binding on successors-in-interest
149 regardless of whether such provisions are included in the relevant transfer documents.

150 Such statements should be addressed to: cyberhygiene@nist.gov

151 Executive Summary

152 Software used for computing technologies must be maintained because there are many in the
153 world who continuously search for and exploit flaws in software. Software maintenance includes
154 *patching*, which is the act of applying a change to installed software – such as firmware,
155 operating systems, or applications – that corrects security or functionality problems or adds new
156 capabilities. *Enterprise patch management* is the process of identifying, prioritizing, acquiring,
157 installing, and verifying the installation of patches, updates, and upgrades throughout an
158 organization.

159 In past perimeter-based security architectures, most software was operated on internal networks
160 protected by several layers of network security controls. While patching was generally
161 considered important for reducing the likelihood of compromise and was a common compliance
162 requirement, patching was not always considered a priority. In today’s environments, patching
163 has become more important, often rising to the level of mission criticality. As part of a zero trust
164 approach to security, it is now recognized that the perimeter largely does not exist anymore, and
165 most technologies are directly exposed to the internet, putting systems at significantly greater
166 risk of compromise. This dynamic applies across all computing technologies, whether they are
167 information technology (IT), operational technology (OT), Internet of Things (IoT), mobile,
168 cloud, virtual machine, container, or other types of assets. Zero trust architectures emphasize
169 business asset-specific security over just protecting a network with assets on it, so patching is
170 vital for reducing risk to those individual assets and determining the assets’ trust status.

171 There is often a divide between business/mission owners and security/technology management.
172 Business/mission owners may believe that patching negatively affects productivity, since it
173 requires scheduled downtime for maintenance and introduces the risk of additional downtime if
174 something goes wrong and disrupts operations. Leadership and business/mission owners should
175 reconsider the priority of enterprise patch management in light of today’s risks. Patching should
176 be considered a standard cost of doing business and should be rigorously followed and tracked.
177 Just as preventive maintenance on corporate fleet vehicles can help avoid costly breakdowns,
178 patching should be viewed as a normal and necessary part of reliably achieving the
179 organization’s missions. If an organization needs a particular technology to support its mission, it
180 also needs to maintain that technology throughout its life cycle – and that includes patching.

181 Leadership, business/mission owners, and security/technology management teams should jointly
182 create an enterprise patch management strategy that simplifies and operationalizes patching
183 while also improving its reduction of risk. This will strengthen organizational resiliency to active
184 threats and minimize business and mission impacts. This publication provides recommendations
185 for enterprise patch management planning.

186 **Table of Contents**

187 **Executive Summary** **iv**

188 **1 Introduction**..... **1**

189 1.1 Purpose and Scope 1

190 1.2 Changes from Previous Versions..... 1

191 1.3 Publication Structure..... 2

192 **2 Risk Response Approaches for Software Vulnerabilities**..... **3**

193 2.1 Risk Responses..... 3

194 2.2 Software Vulnerability Management Lifecycle..... 4

195 2.3 Risk Response Execution 5

196 2.3.1 Prepare to Deploy the Patch..... 5

197 2.3.2 Deploy the Patch 5

198 2.3.3 Verify Deployment 6

199 2.3.4 Monitor the Deployed Patches..... 6

200 **3 Recommendations for Enterprise Patch Management Planning**..... **8**

201 3.1 Reduce Patching-Related Disruptions 9

202 3.2 Inventory Your Software and Assets 10

203 3.3 Define Risk Response Scenarios..... 11

204 3.4 Assign Each Asset to a Maintenance Group..... 12

205 3.5 Define Maintenance Plans for Each Maintenance Group 14

206 3.5.1 Maintenance Plans for Scenario 1, Routine Patching 14

207 3.5.2 Maintenance Plans for Scenario 2, Emergency Patching 14

208 3.5.3 Maintenance Plans for Scenario 3, Emergency Workarounds 14

209 3.5.4 Maintenance Plans for Scenario 4, Unpatchable Assets..... 15

210 3.6 Choose Actionable Enterprise-Level Patching Metrics 15

211 3.7 Consider Software Maintenance in Procurement..... 17

212 **References** **18**

213 **Appendix A— Mappings to NIST Guidance and Frameworks** **19**

214 **Appendix B— Acronyms** **20**

215

216 **1 Introduction**

217 **1.1 Purpose and Scope**

218 The purpose of this publication is to help organizations improve their enterprise patch
219 management planning so that they can strengthen their management of risk. This publication
220 strives to illustrate that enterprise patch management is preventive maintenance for an
221 organization's technology. Adopting this mindset and following the recommendations and
222 suggestions in this document should help organizations in the following ways:

- 223 • Security and technology management and leadership will gain a new understanding of the
224 role of patching in enterprise risk management.
- 225 • The security/technology and business/mission sides of the organization will be able to
226 communicate with each other more effectively regarding patch management and reach
227 consensus on planning.
- 228 • Personnel from the security/technology and business/mission sides of the organization
229 will be prepared to revamp their enterprise patching strategy throughout the entire patch
230 management life cycle.

231 The discussion of patch management technologies is minimal in this publication. However, NIST
232 Special Publication (SP) 1800-31 [1] provides information on using technologies to implement
233 information technology (IT) patch management policies and processes.

234 This publication is intended to apply to all types of computing technologies –including IT,
235 operational technology (OT), Internet of Things (IoT), mobile devices, and cloud computing –
236 and to all types of patchable software – including applications, operating systems, and firmware
237 – on those technologies.

238 **1.2 Changes from Previous Versions**

239 This is the fourth version of NIST SP 800-40. The original SP 800-40, [Procedures for Handling](#)
240 [Security Patches](#) (2002), provided basic information on patching procedures and sources of patch
241 and vulnerability information. SP 800-40 Version 2.0, [Creating a Patch and Vulnerability](#)
242 [Management Program](#) (2005), built on the original by adding content on processes, metrics, and
243 common issues. Although SP 800-40 and SP 800-40 Version 2.0 are primarily of interest from a
244 historical perspective, they address many of the same topics that organizations are still struggling
245 with today.

246 The third version, SP 800-40, Revision 3, [Guide to Enterprise Patch Management Technologies](#)
247 (2013), was written under the assumption that readers already understood the basics of patch
248 management and that what they most needed help with was implementing, configuring, securing,
249 and using enterprise patch management technologies. The latest SP 800-40 version is based on
250 the assumption that, in the overall scope of enterprise patch management, organizations would
251 benefit more from rethinking their patch management planning than their patch management
252 technology. Readers who are particularly interested in enterprise patch management technologies
253 may still benefit from content in Revision 3, although some of it is outdated and there are gaps in
254 its coverage.

255 1.3 Publication Structure

256 The rest of this publication is organized into the following sections and appendices:

- 257 • Section 2 outlines possible risk response approaches for software vulnerabilities and
258 provides a brief overview of the enterprise patch management life cycle.
- 259 • Section 3 presents a set of principles and actionable recommendations that support those
260 principles for enterprise patch management planning.
- 261 • The References section defines the references cited throughout the publication.
- 262 • Appendix A lists the NIST Cybersecurity Framework Subcategories and the SP 800-53
263 controls that are most important for enterprise patch management policies and processes.
- 264 • Appendix B contains an acronym list.

2 Risk Response Approaches for Software Vulnerabilities

This section outlines possible risk response approaches for software vulnerabilities, provides an overview of the software vulnerability management life cycle, and takes a closer look at parts of that life cycle with respect to patching.

2.1 Risk Responses

Patching is one of several ways to respond to risks from software vulnerabilities. This publication references four types of *risk responses* [2]:

1. **Accept:** Accept the risk from vulnerable software as is, such as by relying on existing security controls to prevent vulnerability exploitation or by determining that the potential impact is low enough that no additional action is needed.
2. **Mitigate:** Reduce the risk by eliminating the vulnerabilities (e.g., patching the vulnerable software, disabling a vulnerable feature, or upgrading to a newer software version without the vulnerabilities) and/or deploying additional security controls to reduce vulnerability exploitation (e.g., using firewalls and network segmentation to isolate vulnerable devices, thus reducing the attack surface).
3. **Transfer:** Reduce the risk by sharing some of the consequences with another party, such as by purchasing cybersecurity insurance or by replacing conventional software installations with software-as-a-service (SaaS) usage where the SaaS vendor/managed service provider takes care of patching.
4. **Avoid:** Ensure that the risk does not occur by eliminating the attack surface, such as by uninstalling the vulnerable software, decommissioning devices with the vulnerabilities, or disabling computing capabilities in devices that can function without them.

By default, an organization accepts the risk posed by using its software. Software could have vulnerabilities in it at any time that the organization does not know about, and sometimes previously unknown vulnerabilities are exploited – a zero-day attack. Once a new vulnerability becomes publicly known, risk usually increases because attackers are more likely to develop exploits that target the vulnerable software.

Installing a patch or update or upgrading software to a newer version without the vulnerabilities are the only forms of risk response that can completely eliminate the vulnerabilities without removing functionality. However, immediately patching, updating, or upgrading vulnerable software is sometimes not viable. Examples of why include the following:

- A patch may not be available yet. For example, a vulnerability may be announced before a patch is ready, and it could be days, weeks, or months before the patch is released.
- The vendor may no longer support the vulnerable software, meaning that a patch for it will never be released because the software is at end-of-life.
- The organization may need to wait for a scheduled outage window, perform testing first, update other software that interacts with the software to be patched, or train employees on new features or interfaces.

- 303 • Some patches may be considered a higher priority, so other patches are delayed due to
304 limited resources.
- 305 • The manufacturer may update the software itself on a delayed schedule, such as for
306 devices with human safety implications in a highly regulated sector, because of the
307 extensive testing and certification that must be performed first.
- 308 • The organization may need to comply with specific legal, regulatory, or business
309 requirements. For example, an organization may need to use Federal Information
310 Processing Standards (FIPS)-validated cryptographic modules for protecting data, but the
311 cryptographic modules in the upgraded software are not yet FIPS-validated.

312 Even when patching, updating, or upgrading vulnerable software is viable, organizations can
313 choose to respond to the risk from the vulnerabilities in a different way, such as any of the other
314 risk response examples at the beginning of this section.

315 2.2 Software Vulnerability Management Life Cycle

316 The following describes a basic software vulnerability management life cycle. This life cycle
317 applies to all risk response approaches.

- 318 1. **Know when new software vulnerabilities affect your organization's assets, including**
319 **applications, operating systems, and firmware.** This involves knowing what assets
320 your organization uses and which software and software versions those assets run down
321 to the level of packages and libraries, as well as keeping track of new vulnerabilities in
322 that software. For example, your organization might subscribe to vulnerability feeds from
323 software vendors, security researchers, and the [National Vulnerability Database \(NVD\)](#).
- 324 2. **Plan the risk response.** This involves choosing which form of risk response (or
325 combination of forms) to use and deciding how to implement the risk response. For
326 example, you might choose mitigation, and the implementation could involve mitigating
327 the vulnerability by upgrading the vulnerable software and altering the software's
328 configuration settings.
- 329 3. **Execute the risk response.** This will vary depending on the nature of the selected risk
330 response, but common phases include the following:
 - 331 a. **Prepare the risk response.** This encompasses any preparatory activities, such as
332 acquiring, validating, and testing patches for the vulnerable software; deploying
333 additional security controls to safeguard the vulnerable software; or acquiring a
334 replacement for a legacy device that cannot be patched. It might also include
335 scheduling the risk response and coordinating deployment plans with enterprise
336 change management, business units, and others.
 - 337 b. **Implement the risk response.** Examples of this include distributing and
338 installing a patch, purchasing cybersecurity insurance, deploying additional
339 security controls, and changing asset configurations and state (e.g., software reset,
340 platform reboot). Any issues that occur during implementation should be
341 resolved.

- 342 c. **Verify the risk response.** This step involves ensuring that the implementation has
343 been completed successfully. For patching, this means confirming that the patch
344 is installed and has taken effect. For deploying additional security controls, ensure
345 they are functioning as intended. For risk avoidance, verify that vulnerable
346 devices were decommissioned or replaced.
- 347 d. **Continuously monitor the risk response.** Make sure that the risk response
348 continues to be in place: no one uninstalls the patch, deactivates the additional
349 security controls, lets the cybersecurity insurance lapse, or restarts the
350 decommissioned device.

351 In addition, there are administrative activities occurring throughout the software vulnerability
352 management life cycle, such as audit logging and generating actionable insights and reports.

353 2.3 Risk Response Execution

354 This section takes a closer look at the common phases of executing a risk response, as described
355 in the previous subsection, specifically within the context of patching.

356 2.3.1 Prepare to Deploy the Patch

357 Examples of common steps for preparing to deploy a patch include the following (not necessarily
358 in this order):

- 359 • **Prioritize the patch.** A patch may be a higher priority to deploy than others because its
360 deployment would reduce cybersecurity risk more than other patches would. Another
361 patch may be a lower priority because it addresses a low-risk vulnerability on a small
362 number of low-importance devices.
- 363 • **Schedule patch deployment.** Many organizations schedule patch deployments as part of
364 their enterprise change management activities.
- 365 • **Acquire the patch.** Patches may be downloaded from the internet, built internally by
366 developers or system administrators, or provided through removable media.
- 367 • **Validate the patch.** A patch's integrity should be confirmed before the patch is tested or
368 installed. The patch could have been acquired from a rogue source or tampered with in
369 transit or after acquisition.
- 370 • **Test the patch.** A patch may be tested before deployment. This is intended to reduce
371 operational risk by identifying problems with a patch before placing it into production.

372 2.3.2 Deploy the Patch

373 Patch deployment varies widely based on several factors, including:

- 374 • The type of software being updated (e.g., firmware, OS, application)
- 375 • The asset platform type (e.g., IT, OT, IoT, mobile, cloud, virtual machine [VM],
376 containers)

- 377 • Platform traits, such as managed/unmanaged asset, on-premises or not, virtualized or not,
378 and containerized or not
- 379 • Environmental limitations, such as network connectivity and bandwidth

380 Many aspects of patch deployment are dependent on patch management technologies, which are
381 out of the scope of this publication. At a high level, examples of common steps for deploying a
382 patch include the following:

- 383 • **Distribute the patch.** Distributing the patch to the assets that need to have it installed can
384 be organization-controlled (and occur automatically, manually, or as scheduled) or
385 vendor-controlled, such as delivered from the cloud.
- 386 • **Validate the patch.** As discussed in Section 2.3.1, a patch's integrity can be confirmed
387 before installation.
- 388 • **Install the patch.** Installation can occur in numerous ways, including automatically;
389 manually when directed to do so by a user, administrator, vendor, or tool; as a result of
390 other software being installed or updated; and through the replacement of removable
391 media used by an asset. Some installations require administrator privileges, such as
392 installing firmware patches for a system BIOS.¹ Some patch installations require user
393 participation or cooperation.
- 394 • **Change software configuration and state.** In some cases, making a patch take effect
395 necessitates implementing changes. Examples include restarting patched software,
396 rebooting the operating system or platform on which the patched software runs,
397 redeploying the applications, or altering software configuration settings. In other cases,
398 no such changes are needed.
- 399 • **Resolve any issues.** Installing a patch may cause side effects to occur, like inadvertently
400 altering existing security configuration settings or adding new settings, and these side
401 effects can inadvertently create a new security problem while fixing the original one.
402 Patch installation can also cause operational issues that may necessitate uninstalling the
403 patch, reverting to the previous version of the software, or restoring the software or asset
404 from backups.

405 **2.3.3 Verify Deployment**

406 A patch's deployment can be verified to ensure that it has been installed successfully and taken
407 effect. The robustness of verification can vary a great deal and is largely dependent on an
408 organization's needs.

409 **2.3.4 Monitor the Deployed Patches**

410 In the last phase of the life cycle, the patch's deployment can be monitored to confirm that the
411 patch is still installed. For example, the patch has not been uninstalled by a user or an attacker,

¹ See [NIST SP 800-147, BIOS Protection Guidelines](#) (2011), for additional information on BIOS updates.

412 an older version of the software has not been restored from a backup, and the device has not been
413 reset to a vulnerable factory-default state.

414 Another reason for monitoring the deployed patches is to see if the patched software's behavior
415 changes after patching. As part of a layered security approach to mitigating supply chain risk,
416 this might be helpful at detecting, responding to, and recovering from situations where the
417 installed patch was itself compromised.

3 Recommendations for Enterprise Patch Management Planning

419 Enterprise patch management has been a contentious issue for decades, with personnel from the
420 security and business/mission sides of organizations often having conflicting opinions. For
421 example, many organizations have struggled with balancing the trade-offs between earlier
422 deployment and more testing. Deploying patches more quickly reduces the window of
423 opportunity for attackers but increases the risk of operational disruption because of the lack of
424 testing. Conversely, testing patches before deployment decreases the risk of operational
425 disruption but increases the window of opportunity for attackers. Testing can also consume
426 considerable staff resources, and it still might miss problems.

427 What has made enterprise patch management tougher recently is how dynamic and dispersed
428 computing assets are, as well as the sheer number of installed software components to patch. In
429 addition, patch management processes and technology take different forms depending on the
430 type of assets (e.g., OT, IoT, mobile, cloud, traditional IT, virtual machines, containers). The
431 result is that many organizations are unable to keep up with patching. Patching often becomes
432 primarily reactive (i.e., quickly deploy a patch when a severe vulnerability is being widely
433 exploited) versus proactive (i.e., quickly deploy patches to correct many vulnerabilities before
434 exploitation is likely to occur).

435 Being proactive means doing more work now to reduce the likelihood of incidents in the future.
436 It also means that if a patch fails, that disruption can be managed and remediated on the
437 organization's schedule. Being reactive means that a compromise of an unpatched vulnerability
438 will occur (e.g., a data breach, a ransomware infection, etc.), the organization will have to
439 perform incident response, their reputation may be damaged, and/or they may potentially be
440 fined or sued. As part of incident response efforts, the missing patch will probably need to be
441 installed anyway in addition to other prerequisite recovery actions, such as reverting to a good
442 known state or rebuilding the environment from scratch.

443 What needs to change in many organizations is the perception that an operational disruption
444 caused by patching is harm that the organization is doing to itself, while an operational
445 disruption caused by a cybersecurity incident is harm caused by a third party. While those may
446 be true statements in isolation, they are misleading and incomplete as part of an organization's
447 risk responses. Disruptions from patching are largely controllable, while disruptions from
448 incidents are largely uncontrollable. Disruptions from patching are also a necessary part of
449 maintaining nearly all types of technology in order to avoid larger disruptions from incidents.

450 That being said, security and technology personnel can take steps to reduce the likelihood of
451 patching causing disruptions, as well as direct patching efforts to prioritize the vulnerabilities
452 that are causing the most risk to the organization. Planning these actions necessitates cooperation
453 between the security/technology and business/mission sides of the organization. This section
454 presents actionable recommendations that organizations should implement to improve their
455 enterprise patch management planning, thereby minimizing the potential negatives of patching to
456 operations.

457 The recommendations support the following principles, which organizations should strive to
458 adopt in their enterprise patch management practices:

- 459 • **Problems are inevitable; be prepared for them.** Risk responses, including patching,
460 will never be perfect. Some may inadvertently cause operational problems, for example,
461 but most will not. To improve enterprise patch management, organizations need to
462 change their culture so that instead of fearing problems and thus delaying risk responses,
463 personnel are prepared to address problems when they occur. The organization needs to
464 become more resilient, and everyone in the organization needs to understand that
465 problems caused by patching are a necessary inconvenience that helps prevent major
466 compromises.
- 467 • **Simplify decision making.** Conducting a risk assessment of each new vulnerability in
468 order to plan the optimal risk response for it is simply not feasible. Organizations do not
469 have the time, resources, expertise, or tools to do so. Planning needs to be done in
470 advance so that when a new vulnerability becomes known, a decision can quickly be
471 made about how to respond to it.
- 472 • **Rely on automation.** There is no way that an organization can keep up with patching
473 without automation because of the sheer number of assets, software installations,
474 vulnerabilities, and patches. Automation is also needed for emergency situations, like
475 patching a severe vulnerability that attackers are actively exploiting. Having automation
476 in place gives an organization agility and scalability when it comes to its risk responses.
- 477 • **Start improvements now.** Some of the changes that an organization may need to make
478 might take years to put in place, but that does not mean that other practices cannot be
479 improved in the meantime.

480 While the recommendations in this section are intended to apply to any organization, small
481 organizations may want additional suggestions for enterprise patch management planning. NIST
482 is developing a basic patch planning playbook that will help simplify planning for small
483 organizations. A pointer to the playbook will be added here once the playbook is available.

484 3.1 Reduce Patching-Related Disruptions

485 **Organizations should strive to decrease the number of vulnerabilities introduced into their**
486 **environments.** This shrinks the attack surface and can lower the amount of patching that
487 organizations need to do. Possible methods for decreasing the number of vulnerabilities include:

- 488 • Harden software, such as enforcing the principles of least privilege and least functionality
489 (e.g., deactivating or uninstalling software services, features, and other components that
490 are not needed). For additional information on hardening assets, especially those
491 considered critical, see the NIST publication [Security Measures for “EO-Critical
492 Software” Use Under Executive Order \(EO\) 14028](#) (July 9, 2021).
- 493 • Acquire software that is likely to have fewer vulnerabilities over time compared to other
494 software.
- 495 • Work with software development partners that are likely to introduce fewer
496 vulnerabilities into software over time.

- 497 • Use managed services instead of software when feasible.
- 498 • Select stacks or platforms that are likely to have fewer vulnerabilities over time compared
- 499 to other stacks or platforms (e.g., running software within a small container instead of a
- 500 larger operating system).

501 **Organizations should consider deploying applications in ways that make patching less**
502 **likely to disrupt operations.** One example is to run applications on stacks or platforms where
503 patching is a fundamental part of the deployed technology and is less likely to disrupt operations
504 (e.g., modernizing and running software within cloud-based containers instead of on-premises
505 server operating systems). Another example is to take advantage of existing toolchains that
506 already build applications with updated components and test them before production release.

507 **3.2 Inventory Your Software and Assets**

508 **Organizations should establish and constantly maintain up-to-date software inventories for**
509 **their physical and virtual computing assets, including OT, IoT, and container assets.** This
510 information could be in a single enterprise asset inventory, or it could be split among multiple
511 resources. While a comprehensive inventory of all assets is ideal, it may be impossible to
512 achieve, given the highly dynamic nature of assets and software. A realistic goal is to maintain a
513 close-to-comprehensive inventory by relying on automation to constantly discover new assets
514 and collect up-to-date information on all assets.

515 Without constant updates, inventories will quickly become outdated and provide increasingly
516 inaccurate and incomplete information for patching efforts. At one time, when assets and
517 software were mostly static and were located within static logical and physical perimeters, it was
518 generally considered acceptable to update inventories on a monthly or quarterly basis by
519 performing a vulnerability scan. That model should no longer be used.

520 Constantly updating inventories for all of the technologies and environments in use today
521 requires a combination of automation techniques and tools. Organizations should leverage
522 inventory capabilities built into platforms and assets whenever feasible. For example, APIs built
523 into a cloud-based platform may enable continuous updates of inventory information for the
524 software on that platform, as well as other platform characteristics helpful for patch management
525 purposes. Vulnerability scans and passive network monitoring on local networks can still
526 contribute to asset inventories, especially in terms of asset discovery. If vulnerability scans are to
527 be used for software inventories, they will need sufficient access to the assets (i.e., authenticated
528 scanning) in order to detect changes to their software and other technical characteristics.

529 **Organizations should approach patching from a per-asset perspective. Software inventories**
530 **should include information on each computing asset's technical characteristics and**
531 **mission/business characteristics.** Making decisions for risk responses and their prioritization
532 should not be based solely on which software and software versions are in use. Each asset has
533 technical and mission/business characteristics that should be taken into consideration because
534 they provide context for the vulnerable software running on that asset.

535 The characteristics that an organization should inventory will vary, but the following are
536 examples of possible characteristics to track:

- 537 • The asset's platform type (e.g., IT, OT, IoT, mobile, cloud, VM)
- 538 • The party who administrates the asset (e.g., IT department, third party, end user,
539 vendor/manufacturer, shared responsibility model)
- 540 • The applications, services, or other mechanisms used to manage the asset (e.g., endpoint
541 management software, virtual machine manager, container management software)
- 542 • The asset's network connectivity in terms of protocols, frequency/duration, and
543 bandwidth
- 544 • The technical security controls already in place to safeguard the asset
- 545 • The asset's primary user(s) or interconnected services and their privileges

546 Examples of mission/business characteristics that an organization should track include:

- 547 • The asset's role and importance to the organization, which are contextual and may be
548 hard to define or determine
- 549 • Laws, regulations, or policies that specify how soon a new vulnerability in the asset must
550 be addressed
- 551 • Contractual restrictions on patching (e.g., a highly regulated device can only be patched
552 by its manufacturer after testing and certification)
- 553 • Mission/business restrictions on risk responses for that asset (e.g., an asset can only be
554 rebooted during a quarterly maintenance outage)

555 Tracking technical and mission/business characteristics for each computing asset provides the
556 basis for better decision making regarding risk responses and priorities. The tracked
557 characteristics are also valuable for other enterprise security and technology purposes, such as
558 supporting efforts to shift to zero-trust architectures.

559 3.3 Define Risk Response Scenarios

560 **Organizations should define the software vulnerability risk response scenarios they need to**
561 **be prepared to handle.** Examples of such scenarios include:

- 562 • **Routine patching.** This is the standard procedure for patches that are on a regular release
563 cycle and have not been elevated to emergency status. Most patching falls under this
564 scenario. However, because routine patching does not have the urgency of emergency
565 scenarios, and routine patch installation can interrupt operations (e.g., device reboots), it
566 is often postponed and neglected. This provides many additional windows of opportunity
567 for attackers.
- 568 • **Emergency patching.** This is the procedure to address patching emergencies in a crisis
569 situation, such as a severe vulnerability or a vulnerability being actively exploited. If one
570 or more of the organization's vulnerable assets have already been compromised,
571 emergency patching may be part of incident response efforts. Emergency patching needs

572 to be handled as efficiently as possible to prevent the imminent exploitation of vulnerable
573 devices.

- 574 • **Emergency workaround.** This is the emergency procedure in a crisis situation, like
575 those described above for the emergency patching scenario, to temporarily mitigate
576 vulnerabilities before a patch is available. The workaround can vary and may or may not
577 need to be rolled back afterward. Emergency workarounds are sometimes needed because
578 of issues with a patch. For example, a patch might be flawed and not actually correct a
579 vulnerability, or a patch might inadvertently disrupt the operation of other software or
580 systems. A patch could even be compromised.
- 581 • **Unpatchable assets.** This is the implementation of isolation or other methods to mitigate
582 the risk of systems that cannot be easily patched. This is typically required if routine
583 patching is not able to accommodate these systems within a reasonable time frame.
584 Examples of why an asset may be unpatchable include the vendor not providing patches
585 (e.g., asset is at end-of-life, asset does not support updates) or an asset needing to run
586 uninterrupted for an extended period of time because it provides mission-critical
587 functions. Unpatchable assets need to be included in risk response planning because a
588 new vulnerability in an asset might necessitate a change in the methods needed to
589 mitigate its risk.

590 3.4 Assign Each Asset to a Maintenance Group

591 **Organizations should use the software inventories, technical and business/mission**
592 **characteristics, and risk response scenarios to assign each asset to a maintenance group.** A
593 *maintenance group* is a set of assets with similar characteristics that generally have the same
594 software maintenance needs for each risk response scenario. Maintenance needs include not only
595 patching (e.g., patch schedule, patch testing needs, outage restrictions, level of impact if
596 vulnerable software is compromised) but also any other appropriate forms of mitigation and risk
597 response, such as temporary workarounds used when patches are not yet available. Organizations
598 should define their maintenance groups at whatever they decide the right level of granularity is.

599 Instead of denoting certain assets or types of assets as “exceptions,” there should be maintenance
600 groups for them. If an asset cannot be patched or should not be patched, there is one less option
601 for addressing its vulnerabilities. It still has software maintenance needs, so it should belong to a
602 maintenance group.

603 Here are a few simplified examples of possible maintenance groups:

- 604 • Mobile workforce laptops for standard end users
 - 605 ○ Software to patch: Firmware, operating systems, and client applications for end
 - 606 user devices
 - 607 ○ Outage restrictions: Tolerant to downtime
 - 608 ○ Existing mitigations: Endpoint security controls running on the laptops
 - 609 ○ Level of impact to the organization if compromised: Moderate
- 610 • On-premises datacenter (including servers, network equipment, storage, etc.)

- 611 ○ Software to patch: Firmware, operating systems, and applications for server
 - 612 platforms
 - 613 ○ Outage restrictions: Must adhere to scheduled outage windows for all non-
 - 614 emergency situations
 - 615 ○ Existing mitigations: Network-based security controls restricting access to the
 - 616 assets and security controls running on the assets themselves
 - 617 ○ Level of impact to the organization if compromised: High
 - 618 ● Legacy OT devices
 - 619 ○ Software to patch: None; existing software is no longer supported and cannot be
 - 620 patched
 - 621 ○ Outage restrictions: Must adhere to scheduled outage windows for all non-
 - 622 emergency situations
 - 623 ○ Existing mitigations: Network isolation, physical security controls
 - 624 ○ Level of impact to the organization if compromised: High
 - 625 ● Smartphones for the mobile workforce
 - 626 ○ Software to patch: Operating systems and mobile apps
 - 627 ○ Outage restrictions: Tolerant to downtime
 - 628 ○ Existing mitigations: Mobile device security controls running on the smartphones
 - 629 ○ Level of impact to the organization if compromised: Moderate
 - 630 ● On-premises servers for automated software testing
 - 631 ○ Software to patch: Firmware, server operating systems, virtualization software,
 - 632 server and client guest operating systems, server and client applications
 - 633 ○ Outage restrictions: Usually tolerant to downtime
 - 634 ○ Existing mitigations: Network-based security controls restricting access to the
 - 635 assets, and security controls running on the assets themselves
 - 636 ○ Level of impact to the organization if compromised: Moderate
 - 637 ● Containers with customer-facing applications in the public cloud
 - 638 ○ Software to patch: Container operating systems, application modules
 - 639 ○ Outage restrictions: Highly tolerant to downtime
 - 640 ○ Existing mitigations: Security controls running on the container operating system
 - 641 ○ Level of impact to the organization if compromised: High
- 642 Maintenance groups can also be defined based on other characteristics, like personnel roles (e.g.,
- 643 software developer workstations, system administrator workstations) or device importance (e.g.,
- 644 low-impact IoT consumer devices, OT and IoT devices with life-safety impact).

645 3.5 Define Maintenance Plans for Each Maintenance Group

646 **Organizations should define a maintenance plan for each maintenance group for each**
647 **applicable risk response scenario.** A maintenance plan defines the actions to be taken when a
648 scenario occurs for a maintenance group, including the time frames for beginning and ending
649 each action, along with any other pertinent information.

650 The following subsections discuss what the maintenance plan for each scenario might involve.

651 3.5.1 Maintenance Plans for Scenario 1, Routine Patching

652 **Organizations should consider adopting phased deployments for routine patching in which**
653 **a small subset of the assets to be patched receive the patch first.** These assets act as canaries
654 (i.e., bellwethers) for identifying issues and determining the likely operational impact of the
655 patch. In effect, this is how the patching gets tested. If the canary assets indicate that the patch
656 should have minimal impact, the deployment can expand to more or all of the vulnerable assets.
657 Significant problems can be addressed before the rollout expands, or a different risk response –
658 like a temporary workaround – can be planned instead of the patch while the problems are
659 resolved.

660 For larger routine patch rollouts, especially those that will directly impact the organization’s
661 users, multiple rounds of canary assets could be used. For example, there could be a small first
662 round with technically knowledgeable users (e.g., system administrators, security engineers)
663 followed by a larger second round with “early adopters” across the organization who are willing
664 to try app updates and report any problems that occur. Ideally, the early adopters will be
665 representative of the entire user community who will eventually be using the patch.

666 **Organizations should offer flexibility with how soon routine patches are to be installed,**
667 **while also forcing installation after a grace period has ended.** A routine patch does not
668 necessitate immediate installation, but at some point, patches must be installed to reduce the risk
669 for the entire environment. Forcing installation can be direct, like triggering patch execution, or
670 indirect, like preventing network access for unpatched assets until they are patched.

671 3.5.2 Maintenance Plans for Scenario 2, Emergency Patching

672 **Organizations should consider using the same general approach for emergency patching as**
673 **for routine patching, except with a highly accelerated schedule.** Even under emergency
674 circumstances, it may still be beneficial to first deploy a new patch to a small number of canary
675 assets to confirm that the patch is not corrupted and does not break the software. This period
676 could last a few minutes to a few hours, and the emergency patching itself could occur in the
677 following hours or days, depending on how urgent the emergency is.

678 3.5.3 Maintenance Plans for Scenario 3, Emergency Workarounds

679 **Organizations should plan for the quick implementation of multiple types of emergency**
680 **workarounds to protect vulnerable assets.** Workarounds may require deactivating system
681 functionality or isolating an asset from other assets and having automated mechanisms to apply

682 these changes. Without the processes, procedures, and tools in place to implement workarounds,
683 too much time may be lost, and vulnerable devices may be compromised.

684 **Organizations should plan to replace emergency workarounds with permanent fixes.** Once
685 a permanent fix, such as a patch, is available, the patch will need to be deployed and the
686 workaround removed. Schedules should be set and enforced for both patch deployment and
687 workaround removal.

688 **3.5.4 Maintenance Plans for Scenario 4, Unpatchable Assets**

689 **Organizations should plan to implement multiple types of long-term risk mitigation**
690 **methods besides patching to protect vulnerable assets.** There should be an approved set of
691 methods for each maintenance group, and these methods should have been reviewed and
692 analyzed in advance by security architects/engineers to determine their adequacy in mitigating
693 risk. For example, Section 3 of NIST SP 800-207 [3] describes the use of micro-segmentation,
694 software-defined perimeters, and other risk mitigation methods for isolating assets based on a
695 defined policy.

696 **Organizations should plan on periodically reevaluating their alternatives to patching.** There
697 are two main aspects to this. One is conducting a risk assessment to see if the alternatives to
698 patching are still sufficiently effective at mitigating risk. The other is conducting a cost-benefit
699 analysis to see if the assets provide sufficient value to the organization compared with the
700 additional costs of mitigating, transferring, or accepting the risk of unpatchable assets.

701 **3.6 Choose Actionable Enterprise-Level Patching Metrics**

702 Metrics play several roles in patch management and vulnerability management. A common
703 example is estimating the relative importance of a new vulnerability so that its remediation can
704 be prioritized appropriately, such as something to be addressed by routine patching versus
705 emergency patching or workarounds. There are many free and commercial sources of this
706 information for organizations to leverage.

707 What organizations often find more challenging is identifying meaningful, actionable enterprise-
708 level metrics that they can adopt to monitor and track their progress with patch management to
709 support their continuous improvement and effectiveness program. Moreover, there are numerous
710 audiences for these metrics, potentially including the organization's CEO/Board of Directors,
711 CIO, CISO, mission/business unit leadership, application developers, security and system
712 administrators, and other cybersecurity and IT personnel. Typically, each of these audiences
713 needs a somewhat different set of metrics that corresponds to their role and responsibilities. For
714 example, a Linux system administrator might need metrics on the performance of Linux asset
715 patching, while a business unit's leadership might want to compare the overall effectiveness of
716 their assets' patching with that of the organization's other business units.

717 **Organizations should take advantage of low-level metrics that they already collect when**
718 **developing enterprise-level metrics to capture patching performance.** There is often a wealth
719 of information already available from the inventories of software and assets, especially the
720 assets' technical and mission/business characteristics. Similarly, organizations often have
721 detailed information about the vulnerabilities themselves, such as Common Vulnerability

722 Scoring System (CVSS) scores, threat intelligence about the vulnerabilities being exploited, and
723 other metrics provided by vulnerability management tools that help indicate how important each
724 vulnerability was to mitigate.

725 **Organizations should utilize their existing low-level metrics to develop enterprise-level**
726 **metrics that reflect the relative importance of each vulnerability and patch.** Overly
727 simplistic metrics, such as counting the number of vulnerabilities that the entire organization had
728 and what percentage of them were patched, are not actionable. If you were told that 10 % of your
729 assets were not being patched, what does that actually mean in terms of your organization’s risk?
730 What is the relative importance of each of those assets? If they are the most important assets,
731 then not patching them might be a major problem. If they are the least important assets, then not
732 patching them might indicate reasonable prioritization of limited resources. To look at the assets
733 another way, what is the relative severity of the unpatched vulnerabilities versus the patched
734 vulnerabilities?

735 Metrics that are too simple are generally not actionable because they do not provide enough
736 information. They do not offer the insights into the performance of vulnerability management
737 that are needed to identify the nature of problems and the improvements necessary to address
738 those problems and improve the organization’s vulnerability management performance. Table 1
739 shows a notional example of actionable performance metrics. Each cell provides mitigation
740 metrics based on the relative importance of the assets (low, moderate, or high) and the
741 vulnerabilities (low, medium, high, or critical), with the categories defined by the organization.
742 The metrics in each cell reflect the percentage of assets that were patched by the corresponding
743 maintenance plans’ deadlines, as well as the average (mean) time and median time for patching.

744 **Table 1: Vulnerability Mitigation Time Summary Matrix**

Vulnerability Importance	Asset Importance		
	Low	Moderate	High
Low	By deadline: 64.7 % Average time: 80.4 days Median time: 75.2 days	By deadline: 72.4 % Average time: 34.7 days Median time: 33.7 days	By deadline: 85.0 % Average time: 14.6 days Median time: 8.1 days
Medium	By deadline: 66.5 % Average time: 75.1 days Median time: 70.7 days	By deadline: 68.7 % Average time: 33.2 days Median time: 31.6 days	By deadline: 71.4 % Average time: 12.9 days Median time: 10.5 days
High	By deadline: 68.6 % Average time: 62.1 days Median time: 58.0 days	By deadline: 78.8 % Average time: 26.8 days Median time: 22.1 days	By deadline: 85.5 % Average time: 8.8 days Median time: 8.1 days
Critical	By deadline: 81.4 % Average time: 44.4 days Median time: 41.3 days	By deadline: 92.3 % Average time: 21.2 days Median time: 23.9 days	By deadline: 95.2 % Average time: 5.2 days Median time: 5.1 days

745 With the additional characteristics that an organization has on their assets and vulnerabilities, it
746 can analyze its mitigation time data by platform, business unit, maintenance group, and other
747 characteristics to find the aspects of vulnerability mitigation that are in greatest need of
748 improvement, as well as to set target values for improving those metrics. Analyzing the data by
749 different characteristics can also provide metrics that are more relevant to particular audiences,

750 such as organizational executives, technology leadership, IT operations staff, and compliance
751 professionals.

752 **Organizations should frequently update their low-level metrics and strive for them to be as**
753 **accurate as possible in order to improve the enterprise-level metrics based on them.** If low-
754 level metrics are incorrect, they will negatively impact the enterprise-level metrics calculated
755 from them. For example, if an organization only scans for vulnerabilities monthly, their low-
756 level metrics for the number of vulnerabilities present in their assets would be much smaller than
757 they should be. This would provide a misleading picture of the organization's vulnerability
758 management program. Similarly, collecting low-level metrics through less accurate methods,
759 such as passive (unauthenticated) instead of active (authenticated) vulnerability scans, will
760 generally underreport vulnerabilities and thus skew the higher-level metrics.

761 **3.7 Consider Software Maintenance in Procurement**

762 **Organizations should take software maintenance into consideration when procuring**
763 **software.** Software maintenance is one factor of many that organizations should consider. It is
764 beyond the scope of this publication to provide methodologies for estimating software
765 maintenance costs or factoring software maintenance into procurement decisions. However, the
766 following is a sample questionnaire that an organization could use to help it understand the
767 software maintenance needs of new software that it may procure:

- 768 1. Will you be releasing updates for this software to address vulnerabilities?
- 769 2. Approximately how many patches, updates, and upgrades do you expect to release each
770 year for this software?
- 771 3. For how many years are you committed to correcting vulnerabilities in the software?
- 772 4. Will you release updates on a regular schedule, as needed, or both? If a schedule will be
773 followed, what is that schedule (weekly, monthly, quarterly, etc.)?
- 774 5. Do you have a vulnerability disclosure and incident response program for your software?
- 775 6. When a vulnerability in your software becomes public but a patch, update, or upgrade is
776 not available, how do you recommend that customers protect their computing assets
777 running your software? Will you provide an emergency workaround to prevent
778 vulnerability exploitation while maintaining most or all software functionality?
- 779 7. When your software is patched or updated, how disruptive will that be to the operating
780 software? For instance, will it require restarting the software, rebooting the asset on
781 which the software is running, etc.?

782 **References**

- 783 [1] Diamond T, Kerman A, Souppaya M, Stine K, Johnson B, Peloquin C, Ruffin V, Simos
784 M, Sweeney S, Scarfone K (2021) Improving Enterprise Patching for General IT
785 Systems: Utilizing Existing Tools and Performing Processes in Better Ways. (National
786 Institute of Standards and Technology, Gaithersburg, MD), Draft NIST Special
787 Publication (SP) 1800-31. [https://www.nccoe.nist.gov/projects/critical-cybersecurity-
788 hygiene-patching-enterprise](https://www.nccoe.nist.gov/projects/critical-cybersecurity-hygiene-patching-enterprise)
- 789 [2] Stine KM, Quinn SD, Witte GA, Gardner RK (2020) Integrating Cybersecurity and
790 Enterprise Risk Management (ERM). (National Institute of Standards and Technology,
791 Gaithersburg, MD), NIST Interagency or Internal Report (IR) 8286.
792 <https://doi.org/10.6028/NIST.IR.8286>
- 793 [3] Rose SW, Borchert O, Mitchell S, Connelly S (2020) Zero Trust Architecture. (National
794 Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication
795 (SP) 800-207. <https://doi.org/10.6028/NIST.SP.800-207>
- 796 [4] Souppaya M, Stine K, Simos M, Sweeney S, Scarfone K (2020) Critical Cybersecurity
797 Hygiene: Patching the Enterprise. (National Institute of Standards and Technology,
798 Gaithersburg, MD), [https://www.nccoe.nist.gov/sites/default/files/library/project-
799 descriptions/ch-pe-project-description-final.pdf](https://www.nccoe.nist.gov/sites/default/files/library/project-descriptions/ch-pe-project-description-final.pdf)
- 800 [5] National Institute of Standards and Technology (2018) Framework for Improving Critical
801 Infrastructure Cybersecurity, Version 1.1. (National Institute of Standards and
802 Technology, Gaithersburg, MD). <https://doi.org/10.6028/NIST.CSWP.04162018>
- 803 [6] Joint Task Force (2020) Security and Privacy Controls for Information Systems and
804 Organizations. (National Institute of Standards and Technology, Gaithersburg, MD),
805 NIST Special Publication (SP) 800-53, Rev. 5, Includes updates as of December 10,
806 2020. <https://doi.org/10.6028/NIST.SP.800-53r5>

807 Appendix A—Mappings to NIST Guidance and Frameworks

808 The controls in the NIST SP 800-53, Revision 5, *Security and Privacy Controls for Information*
809 *Systems and Organizations*, control catalog that are most important for enterprise patch
810 management planning are:

- 811 • CM-2, Baseline Configuration
- 812 • CM-3, Configuration Change Control
- 813 • CM-8, System Component Inventory
- 814 • RA-7, Risk Response
- 815 • SI-2, Flaw Remediation
- 816 • SR-2, Supply Chain Risk Management Plan
- 817 • SR-3, Supply Chain Controls and Processes
- 818 • SR-5, Acquisition Strategies, Tools, and Methods

819 The Subcategories from the Cybersecurity Framework that are most important for enterprise
820 patch management planning are:

- 821 • ID.AM-1: Physical devices and systems within the organization are inventoried
- 822 • ID.AM-2: Software platforms and applications within the organization are inventoried
- 823 • ID.AM-4: External information systems are catalogued
- 824 • ID.AM-5: Resources (e.g., hardware, devices, data, time, personnel, and software) are
825 prioritized based on their classification, criticality, and business value
- 826 • ID.BE-3: Priorities for organizational mission, objectives, and activities are established
827 and communicated
- 828 • ID.BE-5: Resilience requirements to support delivery of critical services are established
829 for all operating states (e.g., under duress/attack, during recovery, normal operations)
- 830 • ID.GV-3: Legal and regulatory requirements regarding cybersecurity, including privacy
831 and civil liberties obligations, are understood and managed
- 832 • ID.RA-5: Threats, vulnerabilities, likelihoods, and impacts are used to determine risk
- 833 • ID.RA-6: Risk responses are identified and prioritized
- 834 • ID.SC-1: Cyber supply chain risk management processes are identified, established,
835 assessed, managed, and agreed to by organizational stakeholders
- 836 • PR.IP-1: A baseline configuration of information technology/industrial control systems is
837 created and maintained incorporating security principles (e.g., concept of least
838 functionality)
- 839 • PR.IP-3: Configuration change control processes are in place
- 840 • PR.IP-12: A vulnerability management plan is developed and implemented

841 Appendix B—Acronyms

842 Selected acronyms and abbreviations used in this paper are defined below.

843	BIOS	Basic Input/Output System
844	BYOD	Bring Your Own Device
845	CEO	Chief Executive Officer
846	CIO	Chief Information Officer
847	CISO	Chief Information Security Officer
848	CVSS	Common Vulnerability Scoring System
849	ERM	Enterprise Risk Management
850	FIPS	Federal Information Processing Standards
851	FISMA	Federal Information Security Modernization Act
852	FOIA	Freedom of Information Act
853	IaaS	Infrastructure as a Service
854	IoT	Internet of Things
855	IR	Interagency or Internal Report
856	IT	Information Technology
857	ITL	Information Technology Laboratory
858	NIST	National Institute of Standards and Technology
859	OMB	Office of Management and Budget
860	OS	Operating System
861	OT	Operational Technology
862	PaaS	Platform as a Service
863	PC	Personal Computer
864	SaaS	Software as a Service
865	SAN	Storage Area Network
866	SP	Special Publication