

The attached DRAFT document (provided here for historical purposes) has been superseded by the following publication:

Publication Number: **NISTIR 7966**

Title: **Security of Interactive and automated Access
Management Using Secure Shell (SSH)**

Publication Date: **10/30/2015**

- Final Publication: *Link to publication DOI*
<http://nvlpubs.nist.gov/nistpubs/ir/2015/NIST.IR.7966.pdf> -or-
DOI URL: <http://dx.doi.org/10.6028/NIST.IR.7966>
(the DOI URL is actually the same link as to the 1st one (nvlpubs.nist.gov))
- Related Information on CSRC NISTIR page:
<http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7966>
- Information on other NIST Computer Security Division publications and programs can be found at: <http://csrc.nist.gov/>

The following information was posted with the attached DRAFT document:

NIST Announces the Release of NISTIR 7966, Security of Interactive and automated Access Management Using Secure Shell (SSH)

October 30, 2015

NIST announces the final release of NIST Internal Report (NISTIR) 7966, Security of Interactive and Automated Access Management Using Secure Shell (SSH). The purpose of this document is to assist organizations in understanding the basics of Secure Shell (SSH) and SSH access management in an enterprise, focusing on the management of SSH user keys. It describes the primary categories of vulnerabilities in SSH user key management and recommends practices for planning and implementing SSH access management.

NISTIR 7966 (Second Draft)

**Security of Interactive and
Automated Access Management
Using Secure Shell (SSH)
(Second Draft)**

Tatu Ylonen
Paul Turner
Karen Scarfone
Murugiah Souppaya

NISTIR 7966 (Second Draft)

**Security of Interactive and
Automated Access Management
Using Secure Shell (SSH)
(Second Draft)**

Tatu Ylonen
*SSH Communications Security
Helsinki, Finland*

Paul Turner
*Venafi
Salt Lake City, Utah*

Karen Scarfone
*Scarfone Cybersecurity
Clifton, Virginia*

Murugiah Souppaya
*Computer Security Division
Information Technology Laboratory*

March 2015



U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, Acting Under Secretary of Commerce for Standards and Technology and Acting Director

1 National Institute of Standards and Technology Interagency or Internal Report 7966
2 52 pages (March 2015)
3

4
5 Certain commercial entities, equipment, or materials may be identified in this document in order to describe an
6 experimental procedure or concept adequately. Such identification is not intended to imply recommendation or
7 endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the
8 best available for the purpose.

9
10 There may be references in this publication to other publications currently under development by NIST in
11 accordance with its assigned statutory responsibilities. The information in this publication, including concepts
12 and methodologies, may be used by Federal agencies even before the completion of such companion
13 publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where
14 they exist, remain operative. For planning and transition purposes, Federal agencies may wish to closely follow
15 the development of these new publications by NIST.

16 Organizations are encouraged to review all draft publications during public comment periods and provide
17 feedback to NIST. All NIST Computer Security Division publications, other than the ones noted above, are
18 available at <http://csrc.nist.gov/publications>.

19
20 **Public comment period: *March 4, 2015 through April 3, 2015***
21

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: NISTIR7966-comments@nist.gov

22

Reports on Computer Systems Technology

23 The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology
24 (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's
25 measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of
26 concept implementations, and technical analyses to advance the development and productive use of
27 information technology. ITL's responsibilities include the development of management, administrative,
28 technical, and physical standards and guidelines for the cost-effective security and privacy of other than
29 national security-related information in Federal information systems.

30

31

Abstract

32 Users and hosts must be able to access other hosts in an interactive or automated fashion, often with very
33 high privileges, for a variety of reasons, including file transfers, disaster recovery, privileged access
34 management, software and patch management, and dynamic cloud provisioning. This is often
35 accomplished using the Secure Shell (SSH) protocol. The SSH protocol supports several mechanisms for
36 interactive and automated authentication. Management of this access requires proper provisioning,
37 termination, and monitoring processes. However, the security of SSH key-based access has been largely
38 ignored to date. This publication assists organizations in understanding the basics of SSH interactive and
39 automated access management in an enterprise, focusing on the management of SSH user keys.

40

41

Keywords

42 access control; authentication; automated access management; device authentication; interactive access
43 management; Secure Shell (SSH); user authentication

44

45

46

Acknowledgments

47 The authors wish to thank their colleagues who reviewed drafts of this document and contributed to its
48 technical content.

49

50

Trademark Information

51 All registered trademarks or trademarks belong to their respective organizations.

52

53

54

Table of Contents

55 **1. Introduction 1**

56 1.1 Purpose and Scope1

57 1.2 Audience1

58 1.3 Document Structure1

59 **2. The Basics of Access Management and Automated Access Management 2**

60 **3. The Basics of SSH..... 3**

61 3.1 Protocol Basics.....3

62 3.2 Common SSH Use Cases3

63 3.3 Server Authentication4

64 3.4 Client Authentication5

65 3.4.1 Password Authentication 5

66 3.4.2 Host-Based Authentication 7

67 3.4.3 Kerberos Authentication 7

68 3.4.4 Public Key Authentication..... 9

69 3.4.5 User Authentication Summary.....11

70 **4. Vulnerabilities in SSH-Based Access12**

71 4.1 Vulnerable SSH Implementation.....13

72 4.2 Improperly Configured Access Controls13

73 4.3 Stolen, Leaked, Derived, and Unterminated Keys13

74 4.4 Backdoor Keys13

75 4.5 Unintended Usage.....14

76 4.6 Pivoting14

77 4.7 Lack of Knowledge and Human Errors14

78 **5. Recommended Practices for Management.....15**

79 5.1 SSH Security Policies and Procedures15

80 5.1.1 Secure SSH Implementation15

81 5.1.2 SSH Identity and Authorized Keys16

82 5.2 SSH Key-Based Access Provisioning, Life Cycle, and Termination Processes 17

83 5.3 Establish Continuous Monitoring and Audit Processes19

84 5.4 Inventory and Remediate Existing SSH Servers, Keys, and Trust Relationships20

85 5.5 Automate Processes21

86 5.6 Educate Executive Management22

87 **6. SSH-Based Access Management Planning and Implementation.....23**

88 6.1 Identify Needs23

89 6.2 Design the Solution24

90 6.3 Implement and Test Prototype.....25

91 6.4 Deploy the Solution26

92 6.5 Manage the Solution26

93 **7. Solution Planning and Deployment28**

94 **Appendix A— NIST SP 800-53 Controls Mapping.....29**

95 **Appendix B— Cybersecurity Framework Subcategory Mapping34**

96 **Appendix C— SSH Key Management Tool Selection36**

97	Appendix D— Building Block	40
98	D.1 Description	40
99	D.2 Scenario	40
100	D.3 Security Characteristics.....	40
101	D.4 Approach.....	40
102	D.5 Business Value.....	41
103	D.6 Relevant Standards.....	41
104	Appendix E— Acronyms and Abbreviations	42
105	Appendix F— Glossary	43
106	Appendix G— References	46
107		

108 **1. Introduction**

109 **1.1 Purpose and Scope**

110 The purpose of this document is to assist organizations in understanding the basics of Secure Shell (SSH)
111 and SSH access management in an enterprise, focusing on the management of SSH user keys.

112 **1.2 Audience**

113 This document is for security managers, engineers, administrators, and others who are responsible for
114 planning, acquiring, testing, implementing, and maintaining SSH solutions. Portions of the document may
115 be of interest to executives and SSH users.

116 **1.3 Document Structure**

117 The remainder of this document is organized into the following sections and appendices:

- 118 ■ Section 2 discusses the basics of access management and automated access management.
- 119 ■ Section 3 examines the basics of SSH.
- 120 ■ Section 4 describes the primary categories of vulnerabilities in SSH user key management.
- 121 ■ Section 5 lists recommended practices for SSH-based access management.
- 122 ■ Section 6 explains planning and implementation processes for SSH-based access management.
- 123 ■ Section 7 discusses solution planning and deployment.
- 124 ■ Appendix A provides a mapping to NIST SP 800-53 Revision 4 security controls.
- 125 ■ Appendix B provides a mapping to Cybersecurity Framework Version 1.0 subcategories.
- 126 ■ Appendix C lists criteria for selecting and procuring tools for managing SSH keys.
- 127 ■ Appendix D contains a detailed discussion of a particular SSH access-related problem that is relevant
128 across a variety of industry sectors.
- 129 ■ Appendix E defines selected acronyms and abbreviations used in the document.
- 130 ■ Appendix F defines selected terms used in the document.
- 131 ■ Appendix G lists references.

132 2. The Basics of Access Management and Automated Access Management

133 Controlling access to information systems is critical for information security. Access controls exist on
 134 many levels and use many technologies. The levels include physical restrictions on access to hardware;
 135 logical controls for accessing network interfaces, hardware management ports on servers, virtualization
 136 hypervisors, operating system (OS) user accounts, and information through database systems; and logical
 137 controls implemented by applications.

138 Information security (confidentiality, integrity, and availability) is compromised if controls at any of these
 139 levels fail. Breaches at different levels have different implications. Generally, a breach at the hardware,
 140 hypervisor, or OS level is more serious than a breach at the application level. For example, breaking into
 141 a database account on a server may permit reading, modifying, and destroying any data in a database,
 142 bypassing normal database-level controls. Breaking into a “root” (administrator) account generally
 143 permits doing this to all data on all accounts on the system, plus installing deeply hidden backdoors,
 144 modifying the operating system, corrupting data, or rendering the system unbootable.

145 Most operating systems use user accounts as the primary unit of access control. In this document, a user
 146 account means an OS level user account unless otherwise specified. User accounts correspond to people
 147 (including system administrators) and service accounts are used for running application software or are
 148 used internally by the OS. It is worth noting that many applications implement their own user accounts
 149 that do not correspond to OS level user accounts (they essentially share the service account of the
 150 application); however, such application-level accounts are generally beyond the scope of this document.

151 User accounts may be stored in a centralized repository (e.g., Active Directory [AD] or Lightweight
 152 Directory Access Protocol [LDAP]) or may be configured locally on a system. A user account defined
 153 locally is generally distinct on each system (separate password, separate home directory, etc.), even if the
 154 same account name is used on multiple computers, while user accounts defined in a centralized repository
 155 are often available on more than one computer and share the same home directory (on a networked file
 156 system). Service accounts are very commonly local accounts, and accounts for people are often stored in a
 157 directory.

158 In a very real sense, access control is the essence of information security. Other security technologies
 159 primarily exist to implement and enforce access controls, make it harder to analyze and attack access
 160 control systems, limit the impact of actual breaches, evaluate the current state of protections, detect
 161 suspicious activity, counteract undesired activity, or help analyze what happened after the fact. The
 162 critical balance in information security is between the need to grant access and the need to limit access.
 163 Consequently, access must be provisioned (based on proper justification for that level of access) and it
 164 must be eventually terminated (e.g., when an employee leaves the role that justified the access, when a
 165 client system or application requiring automated access is decommissioned).

166 Access to hosts has become increasingly automated. Examples of this automation include file transfers,
 167 disaster recovery, privileged access management, software and patch management, and dynamic cloud
 168 provisioning. This automation involves transferring data and executing commands, such as having hosts
 169 reconfigure other hosts. Thus, hosts must access other hosts, often with very high privileges.
 170 Unfortunately, there has been little planning and oversight of automated and interactive machine-to-
 171 machine access control. Instead, such access has been added and configured on an ad hoc basis by system
 172 administrators, vendors, and integrators as part of other projects, without formal access control lifecycle
 173 management (e.g., standardized provisioning and termination processes, access token management (e.g.,
 174 periodic password changes)). This publication explores the field of SSH-based access management, with
 175 a strong focus on security issues and how to best address them.

176 3. The Basics of SSH

177 Secure Shell (SSH) is a protocol for securely logging into a remote host and executing commands on that
 178 host (e.g., administrative commands). What distinguishes the SSH protocol from earlier remote
 179 administration protocols, such as telnet, remote shell (rsh), remote login (rlogin), and remote copy (rcp),
 180 is its built-in support for robust security features such as secure user and device authentication and
 181 transmission encryption. SSH has almost completely taken the place of these insecure remote
 182 administration protocols.

183 Today, SSH software is natively included and used as the primary remote administration mechanism for
 184 many operating systems and devices, including Linux, Unix, routers, firewalls, network appliances,
 185 security appliances, and other systems. It is also embedded behind the scenes into a wide variety of
 186 Information Technology (IT), networking, and security technologies, including file transfer, systems
 187 management, identity management, and privileged access management. In addition to manually remotely
 188 connecting to and managing hosts and devices, SSH is used for integrating hosts and automating their
 189 operation.

190 This section examines the basics of Secure Shell (SSH) version 2.0. First, Section 3.1 explores SSH
 191 protocol basics. Next, Section 3.2 discusses the most common use cases for SSH, including the use case
 192 scenario of interest in this publication: user and automated access. Section 3.3 examines SSH server
 193 authentication, while Section 3.4 details SSH client authentication.

194 3.1 Protocol Basics

195 The SSH protocol has a typical client/server architecture. An SSH client application on host A initiates a
 196 connection to an SSH server application on host B. These two hosts negotiate encryption algorithms for
 197 their transmissions, then establish a session key and perform device authentication for the server host
 198 (host B)¹, and finally send client (or user) authentication credentials (e.g., username and password) to the
 199 server. Assuming that this authentication succeeds, an SSH connection is said to be established between
 200 the hosts, ready for use.

201 The current version of the SSH protocol is 2.0. Earlier versions of the protocol have serious known
 202 vulnerabilities that preclude their use. For more information on the formal definition of the SSH protocol
 203 version 2.0, see [RFC4251], [RFC4252], [RFC4253], and [RFC4254]. All references to the SSH protocol
 204 in this publication are to version 2.0 unless explicitly stated otherwise.

205 3.2 Common SSH Use Cases

206 There are three common use cases for SSH:

- 207 ■ **Interactive use.** SSH is used for managing and configuring Unix and Linux computers, networking
 208 equipment, and various other types of hosts remotely. SSH is also used for running applications
 209 remotely (particularly text-based legacy applications).
- 210 ■ **File transfers.** SSH is used as the foundation of the Secure Copy (scp) and Secure File Transfer
 211 Protocol (SFTP) protocols. These protocols are used to transfer files between hosts while leveraging
 212 the security capabilities built into SSH.

¹ The primary purpose of authenticating the server is to prevent man-in-the-middle attacks.

213 ■ **Point-to-point tunneling.** SSH can be used to implement a virtual private network (VPN) tunnel to
 214 protect data transmitted between two hosts. One or both of these hosts may be acting as a gateway for
 215 other hosts behind it.

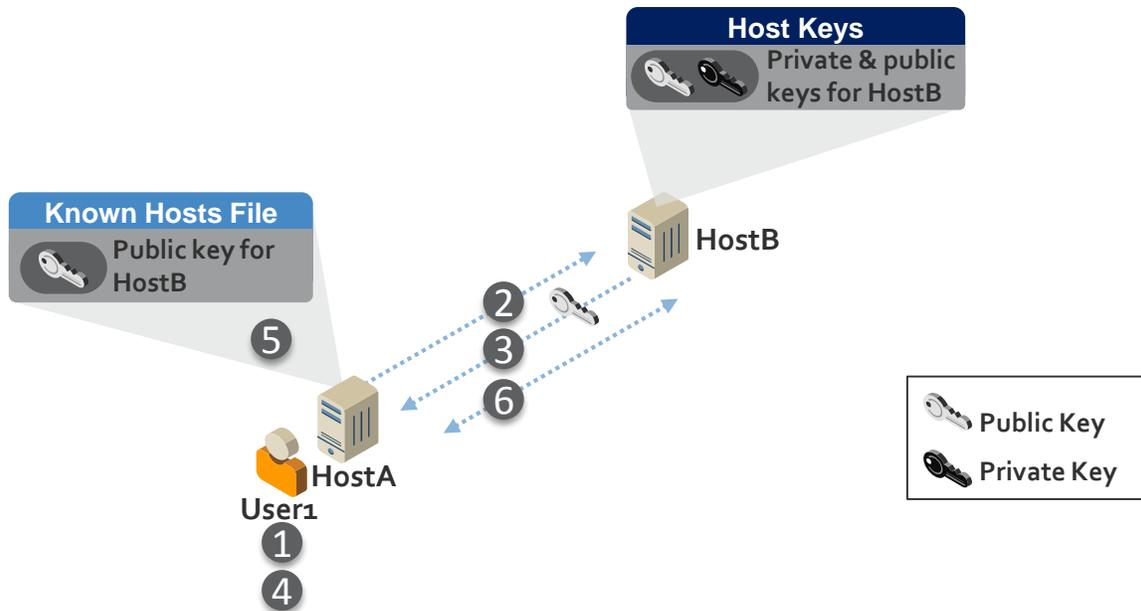
216 This publication covers all of these use cases in the context of user and automated access. Automated
 217 access refers to accessing a host from another host in an automated fashion (without human intervention).
 218 SSH is frequently used for automated access for a variety of purposes, including managing large IT
 219 environments, integrating applications, and provisioning virtual machines in cloud services.

220 Automated access is commonly used with functional accounts, system accounts, service accounts, and
 221 other non-interactive user accounts (sometimes also called non-user accounts). Such accounts are used by
 222 operating systems, server applications (e.g., databases), and other applications for running processes.
 223 Automated access is also frequently used for file transfer functions.

224 Automated access may be unrestricted, allowing any commands to be executed, or may be limited to
 225 specific commands or operations, such as file transfers (perhaps limited to a specific directory).
 226 Organizations should limit automated access so that only the necessary commands can be executed and
 227 only the necessary resources can be engaged.

228 **3.3 Server Authentication**

229 The confidentiality and integrity of SSH rely on strong authentication of both the SSH server and client.
 230 Authentication of the SSH server by the client is performed with public key cryptography via public keys
 231 explicitly trusted by the client or certificates issued by a certification authority—public keys are more
 232 commonly used than certificates.²



233

234 **Figure 3-1: Process for Initially Trusting an SSH Server's Public Key**

² X.509v3 compliant host certificates are a useful alternative for host keys in large environments and make host key rotation much easier.

235 Figure 3-1 illustrates the process for initially trusting an SSH server's public key:

- 236 1. User1 initiates its first login to HostB.
- 237 2. HostA connects to HostB.
- 238 3. HostB sends its public key to HostA.³
- 239 4. The public key sent by HostB is displayed for User1. User1 verifies that it is the correct public
- 240 key for HostB by comparing it against the value received via a different communication channel.
- 241 5. HostA stores HostB's public key in User1's known hosts file along with the name of HostB in
- 242 order to authenticate HostB during future connections without prompting the user to verify that
- 243 the key is the correct public key for HostB.
- 244 6. HostA authenticates HostB using HostB's public key and establishes an encrypted connection
- 245 with HostB.

246 Note: If the User1 account is being used by an automated process, the user who reviews the public key on

247 first connection is typically the administrator for the automated process. To avoid this process, the

248 administrator can prepopulate the known hosts file with HostB's public key.

249 Care must be taken by users and administrators in ensuring that the correct public keys for each server are

250 trusted as known host keys to prevent man-in-the-middle attacks. In addition, a man-in-the-middle attack

251 can also be launched if the attacker is able to get a copy of the server's private key. Wherever possible,

252 users should have limited ability to change known host key configurations. In addition, known hosts files

253 should be hashed to minimize information available to potential attackers.

254 3.4 Client Authentication

255 Client authentication refers to the authentication of interactive users (administrators and other users) or

256 automated processes operating on SSH clients. Authentication occurs for a particular account on the

257 server.

258 The SSH protocol supports several mechanisms for authenticating users, including passwords, host-based

259 authentication, Kerberos, and public key authentication. All these authentication methods fundamentally

260 rely on some secret information, and when used for automated access, this secret information must be

261 stored locally or be otherwise accessible. One or more client authentication methods can be enabled on

262 each SSH server. Each authentication method features pros and cons for security and flexibility—these

263 pros/cons are often different for interactive users and automated process. Organizations must carefully

264 evaluate and select the client authentication method(s) that are approved for use in their environment,

265 disabling others. This section briefly discusses these forms of client authentication, focusing on their

266 relevancy and appropriateness for interactive users and automated processes.

267 3.4.1 Password Authentication

268 There are two kinds of password authentication mechanisms in SSH: basic password authentication and

269 keyboard-interactive authentication. Basic password authentication is a legacy method defined by the

270 SSH protocol standards. Keyboard-interactive authentication is used in most modern environments, and

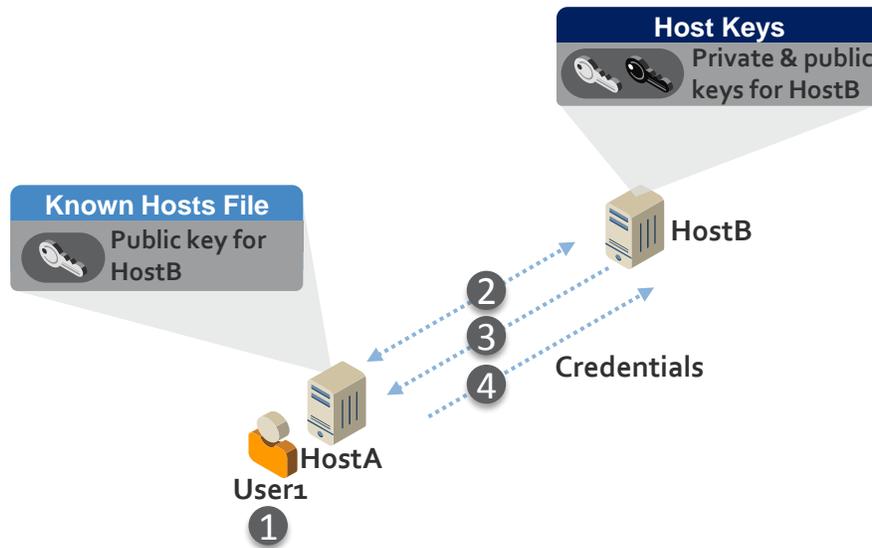
271 can support challenge-response authentication and one-time passwords in addition to traditional password

272 authentication. LDAP may be used in lieu of local credential databases (e.g., password files) to reduce the

273 number of passwords users must remember and manage. With both password and keyboard-interactive

³ An SSH server may have more than one public/private key pair if multiple algorithms (e.g., RSA, DSA, ECDSA) are supported on the server.

274 authentication, usernames, passwords, and challenge responses are sent from the client (HostA) to the
 275 server (HostB) across the encrypted SSH connection. This protects the credentials in transit across the
 276 network but they are still subject to compromise in a man-in-the-middle attack. Password authentication
 277 is more commonly used for interactive users, but less commonly for automated access, although it is
 278 sometimes seen with hard-coded passwords in scripts and management systems.



279

280

Figure 3-2: Password Authentication Process

281 Figure 3-2 illustrates the password authentication process:

- 282 1. User1 starts to login to HostB.
 283 2. HostA authenticates HostB using the HostB public key stored in User1's known hosts file and
 284 establishes an encrypted connection with HostB.⁴
 285 3. User1 is prompted for a username and password and potentially other credentials (for keyboard-
 286 interactive authentication) required by HostB.
 287 4. The credentials are sent to HostB over the encrypted connection. HostB authenticates User1 by
 288 verifying the credentials provided by User1.

289 For interactive users, password authentication provides users mobility, as they can enter their password
 290 anywhere from which they have SSH access. However, if interactive users are required to remember and
 291 manage different passwords for multiple systems, it can create administrative and security challenges.

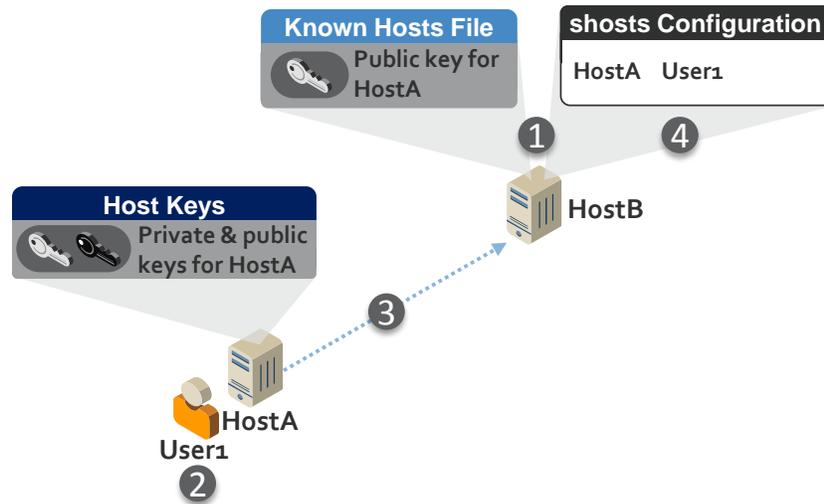
292 Password authentication is generally not recommended for automated processes because it doesn't
 293 provide the level of access control available with other authentication methods, especially public key
 294 authentication.

295 If password authentication is used for interactive users or automated access, the passwords should be
 296 rotated frequently in accordance with the organization's password policy (which should also contain
 297 requirements such as minimum password length, minimum password complexity, etc.)

⁴ This step is described above in Server Authentication. It is included here to promote understanding of when server authentication occurs during the password authentication process.

298 3.4.2 Host-Based Authentication

299 Host-based authentication uses a server host's (HostA) *host key*—the key typically used by clients to
 300 verify the server's identity—to authenticate that server (HostA) to another server (HostB) and to vouch
 301 for the identity of the user (User1) on the client side server (HostA). A configuration file (.shosts) can be
 302 used with any user account on the destination server (HostB) to specify which users on which hosts can
 303 log into that account without further authentication.



304

305

Figure 3-3: Host-Based Authentication

306 Figure 3-3 illustrates host-based authentication:

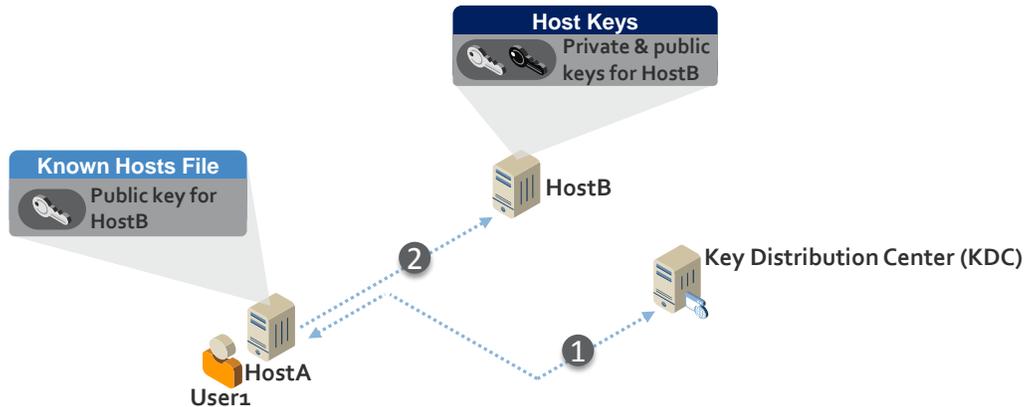
- 307 1. The administrator for HostB places the public key for HostA in the known hosts file on HostB
- 308 and configures the shosts (e.g. ~/.shosts or shosts.equiv) to allow User1 to authenticate from
- 309 HostA.
- 310 2. User1 starts to login to HostB.
- 311 3. HostA authenticates to HostB using the host key for HostA.
- 312 4. HostB confirms in the shosts configuration that User1 is authorized to access the target account
- 313 on HostB from HostA.

314 Host-based authentication does not permit configuring command restrictions—limits on what can be done
 315 on the destination server when accessed. Because of this, it is not recommended for automated access.
 316 Host-based authentication is not recommended for interactive users because it does not present an
 317 interactive login, which is not generally considered a good practice, especially for accounts with elevated
 318 privileges.

319 3.4.3 Kerberos Authentication

320 Kerberos (usually together with an LDAP-based directory, such as Active Directory) implements single
 321 sign-on within a Windows domain or Kerberos realm, and allows user accounts and credentials to be
 322 stored in a centralized directory. A user authenticates to the directory and then is able to access any
 323 account with the same name on SSH-based machines that are joined to the same domain or realm. On one
 324 hand, this can be beneficial for single sign-on and a single point for managing user accounts, including

325 the ability to disable the account centrally remove all access. On the other hand, such single sign-on
 326 implies that once a user has authenticated to an account using Kerberos, it is possible to log in to any
 327 other server that has the same account name and is in the same domain (with Kerberos authentication
 328 enabled) without further authentication. This can easily create lots of unwanted implicit trust
 329 relationships. Another concern is that currently widely used SSH implementations do not support
 330 command restrictions for Kerberos.



331

332

Figure 3-4: Kerberos Authentication

333 Figure 3-4 illustrates Kerberos authentication with SSH:

- 334 1. User1 authenticates to the Kerberos Key Distribution Center (KDC) and receives a ticket which
- 335 HostA stores for future authentications.
- 336 2. User1 goes to connect to HostB. HostA provides the ticket to HostB, which it uses to authenticate
- 337 User1.

338 Kerberos includes a user's IP address(es) within authentication tickets to ensure that these tickets cannot
 339 be copied and reused by an attacker on another system. This feature of Kerberos prevents SSH man-in-
 340 the-middle attacks. However, in environments where NAT firewalls are used, access will be denied when
 341 attempting to access a host on the other side of a firewall. In order to enable access across NAT firewalls,
 342 Kerberos must be configured to allow tickets that do not include IP addresses, which opens the
 343 vulnerability of tickets to being copied and reused.

344 To facilitate authentication for automated processes, many SSH Kerberos implementations provide the
 345 option for credentials to be stored in a file on the client called a *keytab file*. This enables automated logon
 346 without user intervention. Access to keytab files should be restricted to prevent credentials from being
 347 compromised and used for unauthorized access from the same system or a different system.

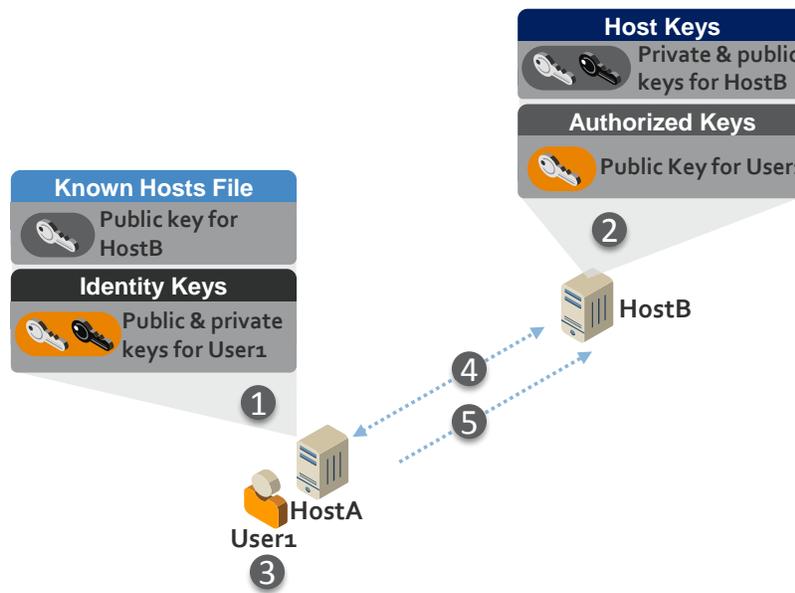
348 For interactive users, Kerberos is best suited for environments where users require access to several
 349 servers, because it provides more secure authentication than password authentication, single sign-on for
 350 users, and single point of account management (enabling/disabling, etc.). However, controls must be put
 351 in place to limit implicit access and ensure that users only have intended access.

352 Automated processes should generally have limited, explicit access to a limited number of specific hosts.
 353 The exception may be management systems that require automated access to a large number of hosts. In

354 practice, Kerberos is rarely used for automated processes and is not recommended due to the risks of
 355 implicit access and the lack of command restrictions.

356 3.4.4 Public Key Authentication

357 Public key authentication in SSH uses user keys or certificates to authenticate a connection – with user
 358 keys being the most commonly used method. Such keys can be configured for both interactive users and
 359 automated processes, and they authorize the user or process to access a user account in an information
 360 system. An interactive user or automated process on an SSH client (HostA) has a user key called an
 361 *identity key*, typically an RSA or DSA private key, and the server (HostB) must have the corresponding
 362 public key configured as an *authorized key* for a user account to which access will be provided. Any user
 363 or process in possession of the identity key is then allowed to log into that user account on the server and
 364 perform actions under the privileges configured for the account.



365

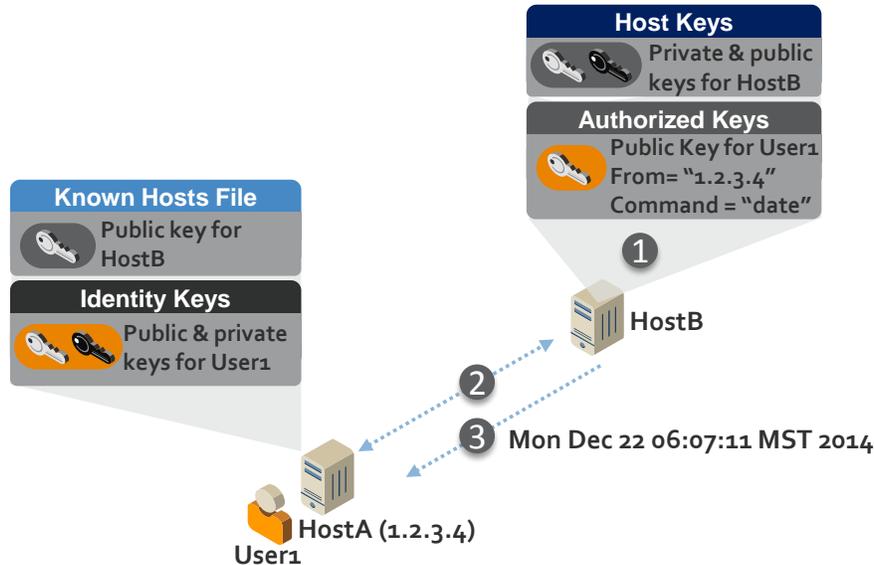
366

Figure 3-5: Public Key Authentication

367 Figure 3-5 illustrates SSH public key authentication:

- 368 1. User1 or an administrator generates an identity key (and corresponding public key) for User1.
- 369 2. The HostB administrator stores User1's public key as an authorized key in the User1 account on
- 370 HostB.
- 371 3. User1 starts to login to HostB.
- 372 4. HostA connects to HostB and attempts to authenticate User1 to HostB using User1's identity key.
- 373 5. HostB authenticates User1 using the authorized key stored in User1's account on HostB.

374 Many SSH implementations support configuring restrictions for authorized keys. These may be used for
 375 limiting what can be done on the server using the key (command restrictions) and for limiting the IP
 376 addresses from which the key can be used (source restrictions).



377

378

Figure 3-6: Public Key Authentication with Source and Command Restrictions

379

Figure 3-6 illustrates public key authentication with source and command restrictions:

380

1. The administrator for HostB configures the authorized key for User1 to only allow authentication to HostB from address 1.2.3.4 and run the *date* command once authenticated.

381

2. User1 authenticates to HostB from HostA (IP address 1.2.3.4).

382

3. HostB executes the *date* command and then terminates User1's connection.

383

384

An important advantage of public key authentication is that it does not create implicit trust relationships, only expressly-defined trust relationships, and the permitted access can be reliably determined by inspecting the destination host.⁵ This is very important for being able to clearly determine and audit who can access each system and account. However, if the deployment of authorized keys is not controlled and tracked, users can create unauthorized trust relationships and access that can be exploited.

385

386

387

388

389

For interactive users, the identity key is usually stored on a smartcard or in a passphrase-protected file on a client device. If the identity key is protected by a passphrase, it is encrypted by a key derived from the passphrase. When SSH user keys are used for automated access, however, the identity key is usually stored as an unencrypted file (with no passphrase) in the file system.⁶ Given that the files grant access to servers, they contain sensitive data.

390

391

392

393

394

Public key authentication is the recommended authentication mechanism for automated access with SSH due to the command and source restrictions that are available and the ability to more readily prevent implicit trust. Public key authentication is by far the most frequently used method of SSH automated access as of this writing.

395

396

397

⁵ An exception is OpenSSH's proprietary certificate authentication, which does not allow reliably auditing who can access a host by inspecting just that host. There is no hardened Certificate Authority solution for OpenSSH, and most environments have no systematic tracking or directory of issued OpenSSH certificates. Thus if the host trusts an OpenSSH certificate signing key, it is often not possible to reliably determine who is able to access the host.

⁶ There is no one present to supply the passphrase for decrypting the identity key for automated access by processes. Hard-coding the passphrase in scripts would provide little additional security (see also NIST SP 800-53 IA-5(7)), and obtaining it from a vault in a script would also provide limited additional security and would be quite a maintenance burden.

398 **3.4.5 User Authentication Summary**

399 There are many considerations to take into account when selecting authentication methods for SSH across
 400 an enterprise. Table 3-1 summarizes the major security and operational considerations for the user
 401 authentication methods described in this section in commonly used SSH implementations. Generally, the
 402 goal should be to select a single method for all authentications or one method for all interactive users and
 403 one for all automated processes. However, this is not always possible based on the options available in
 404 deployed SSH systems and software.

405 For automated processes, public key authentication is the recommended method based on its additional
 406 security features. For interactive users, public key authentication with smartcards provides the best
 407 security. For file-based public key authentication with interactive users, if passphrase strength for
 408 interactive user identity keys stored in files (instead of smartcards) cannot be enforced, weak passphrases
 409 may put identity keys at risk of compromise. In these cases, password or Kerberos authentication, where
 410 password strength can be enforced, may provide higher security. The use of keytab files with Kerberos for
 411 interactive users is not recommended.

412 If multiple authentication methods will be enabled, it is critical to put controls in place to prevent
 413 unintended access. For example, if Kerberos is selected for interactive users and public key authentication
 414 for automated processes, it is critical to ensure that the names of system accounts for automated processes
 415 are not the same as accounts in the central directory. And, conversely, it is also critical that users be
 416 prevented from configuring authorized keys that would allow them circumvent controls enforced via
 417 Kerberos.

418 **Table 3-1: Comparing Authentication Methods**

	Password	Host-Based	Kerberos	Public Key
Supports Portability (User not required to carry credential from system to system)	Yes	Yes	Yes	Yes/No ⁷
Provides Single Sign-On	No	No	Yes	Yes
Enforces Command Restrictions	No	No	No	Yes
Prevents Man-in-the-Middle	No	Yes	Yes	Yes
Minimizes Implicit Access	Yes/No ⁸	Yes	No	Yes
Supports NAT Environments	Yes	Yes	No	Yes ⁹

⁷ If the identity key for an interactive user is stored on a smartcard, the user can easily and securely carry it from system to system. If it is stored in a passphrase-protected file, the user must copy the file to each client from which he or she accesses SSH servers.

⁸ When password or keyboard-interactive authentication is used with LDAP, implicit access can occur when an account on a host unexpectedly has the same name as an account in the LDAP directory.

⁹ Source restrictions must be configured to use the address on the server side of the NAT device.

4. Vulnerabilities in SSH-Based Access

Because SSH is the primary secure access method used for administration and automated processes on mission critical systems, its security is crucial. The systems managed via SSH include Unix and Linux systems, routers, firewalls, security appliances, Unix partitions on mainframes, and other network devices. The privileges granted to users and automated processes via SSH are typically elevated—often root level—privileges.¹⁰ A number of vulnerabilities arise—both for users and automated processes—if proper provisioning, termination, and monitoring processes are not implemented.

However, the security of SSH-based automated access, and even interactive access, has been largely ignored to date. Many organizations don't even know how many SSH keys they have configured to grant access to their information systems or who has copies of those keys. The accounts associated with these keys often grant far more access than is actually needed, such as allowing execution of any command or transfer of files to any directory. Also, in many organizations, system administrators configure new keys without any approvals or coordination, and may use them to circumvent auditing of privileged access and maintenance. Some large enterprises have hundreds of thousands or even millions of SSH user keys on their systems for automated access, which often provide many more entry points onto servers than the interactive user accounts do. Also, a sizable percentage of these keys typically grant access to administrative/root accounts or sensitive accounts, such as those storing database files or critical software.

A closely related issue is the trust relationships that these keys establish within and between systems, even between organizations. Some of these trust relationships may be undesirable or violate policies, such as leading from development and test systems into production systems, or crossing from a low-impact system to a high-impact system without requiring any additional authentication. Most importantly, SSH key-based trust relationships can enable an attacker who compromises one system to quickly move (or pivot) from one system to another and spread through an organization—individual trust relationships often collectively form webs of trust across an organization's systems.

Although the security implications of poor SSH key management have been known for some time, the scope of the problem for automated access has not been widely understood or acknowledged. For example, most organizations do not have SSH key management or SSH-based automated access as part of their assessment programs, even though for many organizations SSH-based automated access has already become central to their identity and access management operations.

This section describes the primary categories of vulnerabilities in SSH-based interactive and automated access—with a particular focus on user keys used to grant access in public key authentication. The recommendations presented in subsequent sections of this document are intended to address these vulnerabilities. The categories of vulnerabilities described in this section are as follows:

- Vulnerable SSH implementation
- Improperly configured access controls
- Stolen, leaked, derived, and unterminated SSH user keys
- Backdoors (unaudited user keys)
- Unintended usage of user keys
- Pivoting

¹⁰ Although access to root accounts via SSH is generally not allowed, administrators will leverage sudo and other tools to elevate their privileges for many operations once logged in via SSH.

458 ■ Lack of knowledge and human errors

459 **4.1 Vulnerable SSH Implementation**

460 An SSH server or client implementation could have vulnerabilities that allow it to be exploited in order to
461 gain unauthorized access to communications or systems. These vulnerabilities could be any of the
462 following types:

463 ■ Software flaws in the SSH implementation (i.e., coding errors)

464 ■ Configuration weaknesses (for example, allowing the use of weak encryption algorithms)

465 ■ Protocol weaknesses (for example, supporting the use of SSH version 1)

466 **4.2 Improperly Configured Access Controls**

467 In its role of enabling administration of systems via elevated privileges—including root—SSH is highly
468 susceptible to enabling unauthorized access due to improperly configured access controls. Most SSH
469 implementations provide a broad number of options for configuring and controlling access, some of
470 which are provided by the SSH software and others through components with which the SSH software
471 integrates, including the underlying OS, pluggable authentication module (PAM), Kerberos, and other
472 security components. Although powerful, improperly configured access controls can inadvertently enable
473 SSH access directly to the root account, unauthorized access to other accounts due to implicit access
474 enabled via Kerberos, the ability to elevate privileges by manipulating the environment, and other access-
475 based vulnerabilities. An administrator may, for example, properly configure SSH but incorrectly
476 configure PAM or not sufficiently limit permissions at the OS for the account being accessed.
477 Furthermore, each OS, appliance, device, or application that supports SSH features different options and
478 methods for controlling access, further complicating efforts to ensure security.

479 **4.3 Stolen, Leaked, Derived, and Undermined Keys**

480 Stolen, leaked, derived, and undermined identity keys pose a similar problem to stolen, leaked, derived,
481 and undermined interactive user account credentials. Anyone who may have obtained a copy of an
482 identity key—by copying it from a host, by accessing a backup, by having malware harvest keys, by
483 performing factoring to derive a key, etc.—may use that key to attempt to gain unauthorized access to a
484 user account on one or more servers in the organization. Once access to a user account has been gained, it
485 is generally possible to access and modify any data for that user account—including reading and
486 modifying the memory of processes running under that user account, and modifying any executable
487 programs owned by that user account.

488 **4.4 Backdoor Keys**

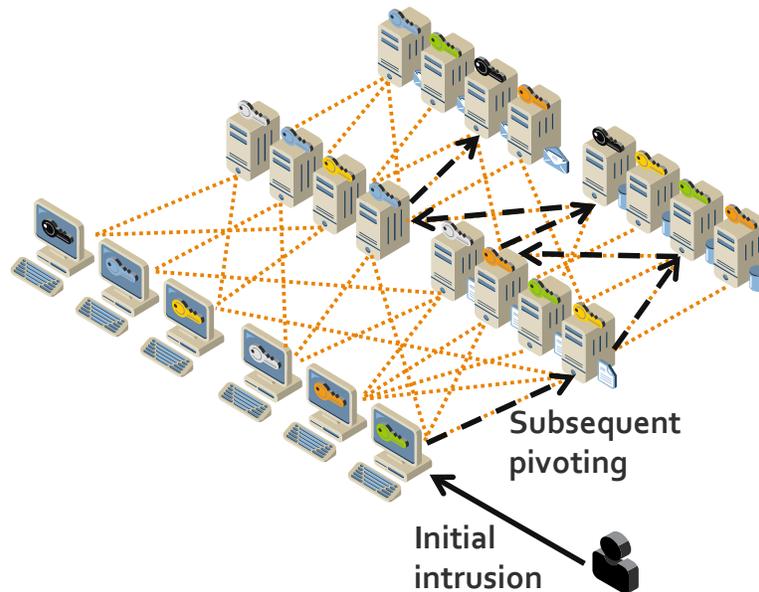
489 Many organizations mandate that all privileged access to their servers take place through a privileged
490 access management system that records all actions performed. Unfortunately, SSH public key
491 authentication can be used to create a “backdoor” that bypasses the privileged access management
492 system. This is done as easily as generating a new key pair and adding a new authorized key to an
493 authorized keys file; authorized keys files are often not audited, so an added key may remain unnoticed
494 for years. The corresponding identity key can then be used to log into the server using any SSH client
495 without the privileged access management system recording the ensuing activity.

496 **4.5 Unintended Usage**

497 Users may, intentionally or unintentionally, use identity keys for purposes for which they were not
 498 intended. An example is using an identity key that was only intended to be used for automated file
 499 transfers to tunnel traffic (thus concealing it from network security controls).

500 **4.6 Pivoting**

501 Malware can be engineered to use SSH keys to spread when automated access is allowed. The mesh of
 502 interactive and automated access relationships is so dense in many cases that it is likely that an attack can
 503 spread (pivot) to most servers in an organization after penetrating the first few servers, especially if other
 504 attack vectors are used to escalate privileges. See Figure 4-1 for an illustration of pivoting.



505

506

Figure 4-1: Pivoting Enabled by Chained SSH Trust Relationships

507 **4.7 Lack of Knowledge and Human Errors**

508 One of the greatest ongoing challenges to the security of SSH-based systems is the potential for human
 509 error due to the complexity of SSH management and the lack of knowledge many administrators have
 510 regarding secure SSH configuration and management. An authorized key may inadvertently be deployed
 511 incorrectly on a host, such as to a root account instead of a regular user account, thus granting
 512 unnecessary privileges. People are known to make human errors when manually setting up new trust
 513 relationships. Such errors can go undetected for years. Also, some key setups involve thousands of hosts,
 514 and while it is easy to miss one or more hosts when copying an authorized key to so many hosts
 515 manually, debugging such errors can be very time consuming. Also, when manually fixing problems,
 516 administrators are likely to just copy the missing keys to the proper accounts without, for example,
 517 checking whether they have accidentally been copied to the root account.

518

5. Recommended Practices for Management

Effectively securing SSH access consists of defining clear policies and implementing management, operational, and technical security control processes to address already deployed SSH systems, privileges, and user keys, and ensure that new SSH systems and user keys are properly deployed and tracked. The objective of management is, as economically as possible, to improve security by following the NIST SP 800-53¹¹ recommended practices for access control with respect to interactive and automated access using SSH.

Additionally, operational processes can be optimized by automating SSH user key setups and removals and related approval, documentation, monitoring, and audit processes. In particular, it is possible to integrate a key management system with a ticketing or change tracking system. If requests for provisioning interactive or automated access are made using a predefined template, a key management system can automatically do the provisioning once the request has been approved, thus reducing labor. Certain provisioning requests may touch thousands of hosts, and some organizations are known to use person-years annually on manual provisioning.

An SSH access management project may also include either selection and acquisition of software tools to aid in the process or development of automatically or manually executed scripts.

5.1 SSH Security Policies and Procedures

Policies and procedures play a critical role in SSH security by establishing consistent baseline requirements across the diverse systems and environments where SSH is deployed, including rapidly evolving cloud environments. The definition of policies should clearly spell out roles and responsibilities in order to prevent misunderstandings that result in security lapses and to ensure accountability. It is critical to educate all SSH stakeholders (system administrators, security professionals, business application owners, etc.) on SSH security policies and processes.

5.1.1 Secure SSH Implementation

Although the hardening of SSH implementations is outside the scope of this document, it is worth noting a few areas where policies and procedures should be defined to mitigate the significant vulnerabilities that can emerge due to poor implementation and management. Other NIST publications and industry standards should be consulted as references for defining policies in the following areas:

- Only enabling SSH server functionality on systems where it is absolutely required.
- Keeping SSH server and client implementations fully up to date across all systems.
- Hardening SSH server and client implementations, including disabling SSH v1 protocol, disabling unapproved authentication methods, preventing implicit access by limiting SSH accessible accounts and groups (including root), disabling port forwarding, limiting access to environment variables, using approved ciphers, properly configuring supporting subsystems (e.g., PAM), and enforcing SSH inactivity timeouts.¹²

¹¹ <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf>

¹² Unfortunately, it is not always possible to harden SSH implementations (e.g., on appliances and embedded devices). In these cases, compensating controls can be used to mitigate the vulnerabilities, such as implementing virtual private network (VPN) tunnels to encapsulate the potentially vulnerable SSH traffic.

- 555 ■ Enforcing least privileged access for all SSH-accessible accounts and groups, especially for
556 automated processes and remote access.

557 5.1.2 SSH Identity and Authorized Keys

558 SSH user keys are security-sensitive configuration information for SSH clients and servers, and their
559 misconfiguration, improper disclosure, or modification may expose servers, applications, and information
560 to unintended access or vulnerabilities. The following policies and procedures are recommended:

- 561
- 562 ■ **Minimum Key Lengths and Approved Algorithms:** To prevent factoring by an attacker, user keys
563 must conform to organizational standards for minimum key lengths used with approved algorithms.
564 NIST SP 800-131A¹³ currently provides recommendations for minimum key lengths that are
565 considered secure in conjunction with various algorithms. Updates to NIST SP 800-131A should be
566 monitored to ensure that the latest recommendations for key lengths are being followed. Note that
567 some older SSH implementations may not support the key lengths recommended by NIST SP 800-
568 131A. Clear criteria should be defined for exceptions where the use of smaller key lengths is allowed.
- 569 ■ **Cryptoperiods (Maximum Key Lifetime):** To minimize the risk of key compromise due to
570 administrative turnover or improper key management by users, the maximum time (cryptoperiod) that
571 an identity key may be used before replacement with a new key should be specified. NIST SP 800-
572 57¹⁴ provides recommended cryptoperiods for public/private authentication keys. Cryptoperiod
573 policies may take into account the potential for compromise based on types of use, including identity
574 keys used for users versus automated processes, the criticality of systems that are being accessed, and
575 the environment where identity keys are stored (e.g., mobile laptops versus smartcard).
- 576 ■ **Identity Key Access Control:** To prevent compromise of a key by an authorized user, access
577 controls on identity keys should be configured to restrict access to the interactive user or automated
578 process to which they have been assigned. For automated processes, policies should define guidelines
579 for assigning access to administrative staff responsible for managing identity keys.
- 580 ■ **Identity Key Passphrases:** To protect against compromise, identity keys assigned to interactive
581 users should be protected by a passphrase. Standards for passphrase strength and changing
582 passphrases each time a key is replaced (at the end of the cryptoperiod) should be defined.
- 583 ■ **Identity Key Duplication:** Identity keys should not be duplicated (copied to other systems) for
584 interactive users and automated processes. If duplication is allowed for interactive users, guidelines
585 should be provided for the acceptable locations where keys can be copied and used.
- 586 ■ **Authorized Key Access Controls:** Access controls on authorized keys should ensure that non-
587 superusers cannot install new authorized keys for user accounts they use. Such new authorized keys
588 can create backdoors, bypass privileged access auditing systems, grant permanent access, or grant
589 access to others, without regards to controls on non-local access and authorization boundaries. In
590 practice this means “locking down keys”, i.e., moving authorized keys to superuser-owned locations
591 that are not writable to non-superusers. It is strongly recommended that authorized keys be moved to
592 protected locations where ordinary users cannot add new or change existing keys, as well as
593 configuring SSH servers to only look for keys from those locations. This is necessary for the
594 following reasons:
- 595 ○ Maintaining control of who can access what information

¹³ <http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf>

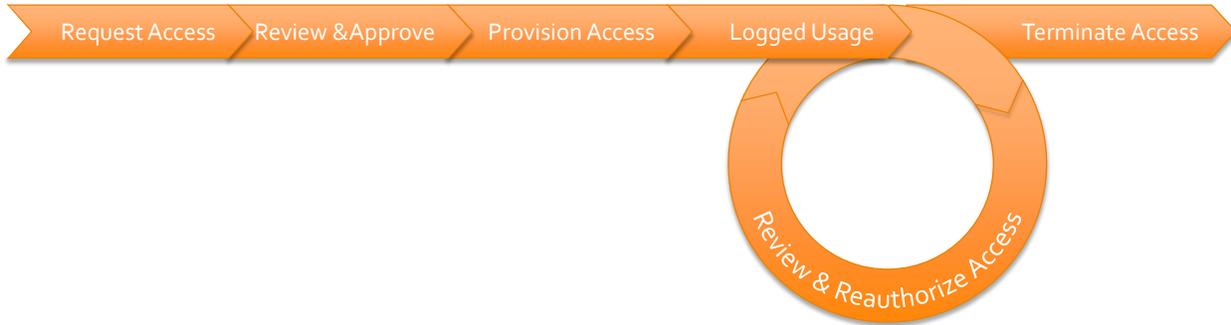
¹⁴ <http://csrc.nist.gov/publications/PubsSPs.html>

- 596 ○ Properly identifying users accessing the system
- 597 ○ Preventing a user from accessing the system after his/her account is terminated
- 598 ○ Ensuring that users cannot bypass privileged access systems
- 599 ○ Controlling remote access
- 600 ○ Enforcing authorization boundaries
- 601 ■ **Authorized Key Command Restrictions:** Authorized keys used for automated processes should be
- 602 configured with command restrictions unless special approval is given. Command restrictions are
- 603 particularly important for keys used for authorizing remote file transfers to avoid accidentally
- 604 permitting remote terminal access and remote execution of commands. For interactive users,
- 605 command restrictions used in conjunction with scripts may be used to limit the commands available
- 606 to users.
- 607 ■ **Authorized Key Source Restrictions:** Authorized keys used for automated processes should be
- 608 configured with source restrictions unless special approval is given, i.e., restrictions on the client IP
- 609 addresses from which the keys can be used. This prevents an attacker who compromises an identity
- 610 key from using it elsewhere on the network.
- 611 ■ **Replacement on Compromise or Reassignment:** SSH keys should be changed when a compromise
- 612 is suspected. In addition, identity keys used for automated processes should be changed prior to
- 613 cryptoperiod expiration whenever an administrator responsible for managing the keys for one or more
- 614 automated processes has been reassigned or terminated.
- 615 ■ **Usage Logging:** All SSH servers should be configured to log key fingerprints for access based on
- 616 SSH authorized keys. This is necessary for performing the log analysis needed for continuous
- 617 monitoring and forensic investigations.
- 618 ■ **Pivot Prevention:** Accounts should not be configured with both incoming and outgoing identity key-
- 619 based trust relationships unless expressly needed (for example, an approved jump server). For
- 620 example, if an authorized key for IdentityKey1 is configured for account User1 on Server1 and that
- 621 same account has another identity key enabling it to authenticate to account User1 on Server2, an
- 622 attacker who gains access to IdentityKey1 can now pivot from Server1 to Server2. Incoming and
- 623 outgoing trust relationships on a server should be configured in separate accounts to prevent pivoting.
- 624 If an organization has needs for limited pivoting, these implementations should be approved on a
- 625 case-by-case basis.
- 626 ■ **No Environment Crossing:** SSH trust relationships should not cross release environment (dev, QA,
- 627 prod) in order to prevent a compromise in a “lower” development or QA environment spreading to
- 628 production environments.

629 5.2 SSH Key-Based Access Provisioning, Life Cycle, and Termination Processes

630 Provisioning and configuring access to an account for interactive users and automated processes should
 631 be a judged decision, balancing the need for access against the risks, and should include consideration of
 632 the level of access required. It is not acceptable for any user or system administrator to grant user account
 633 access to other users or processes without proper approval.

634 In a controlled provisioning and life cycle process, provisioning interactive or automated access requires
 635 the phases depicted in Figure 5-1:



636

637

Figure 5-1: Controlled Provisioning and Life Cycle Process

638 ■ **Request:** Submitting a request for establishing access to one or more accounts on one or more servers
 639 together with justification for the access. The request for automated access should specify the
 640 command that is authorized to be executed (or in exceptional cases, interactive command line access).
 641 Furthermore, a request should identify the business process, application, or information system that
 642 the request relates to and the owner/responsible person for the trust relationship if not the owner of
 643 the business process, application, or information system. The request should also specify the source
 644 account(s) and host(s) from which access is allowed or the key to be used for access. Ideally, an
 645 existing change control system is used for submitting a filled request template.

646 ■ **Approval:** Reviewing and approving the request based on consideration of security impact,
 647 destination account and server, authorization boundary crossing, and remote access considerations.
 648 Ideally, the change control system is used for approving the change in large enterprises or other
 649 organizations that need centralized, automated mechanisms for managing keys. The request should, at
 650 a minimum, be reviewed and approved by the official responsible for the security of the information
 651 system to which it grants access. Access grants create connections between systems and may need to
 652 be documented in security policies, plans, or architecture documents as system interconnections.
 653 Access grant approvals should be recorded in a database to enable them to be used for continuous
 654 monitoring and audits. A controlled process with proper documentation of approvals is necessary for
 655 being able to detect access grants by unauthorized users (e.g., unapproved authorized keys) during
 656 continuous monitoring. Without documented approvals, it will not be possible to audit that all
 657 existing authorized keys have been approved. Documenting why each key is authorized is also
 658 important for being able to remove keys when no longer needed to properly terminate access.

659 ■ **Provisioning:** Implementing the request, creating a key pair if needed, and configuring the authorized
 660 key(s) and identity key(s). SSH key-based access provisioning should be limited to authorized
 661 individuals; ideally, it is performed by a central team that ensures consistent management across all
 662 systems. To improve efficiency and reduce errors, the implementation takes place automatically to
 663 reduce resources, reduce the need for privileged access, and eliminate manual configuration errors.

664 ■ **Usage Logging:** Logging usage to facilitate reauthorizations (e.g., accounts and authorized keys that
 665 are not used for a long period of time should likely not be reauthorized), forensic investigations, and
 666 audits. Logs should be retained for a sufficient amount of time to enable forensic or operational
 667 investigations at a later date.

668 ■ **Review and Reauthorization:** Regular reviews and reauthorization of SSH key-based access by
 669 individuals responsible for the security of systems and data being accessed. An organization should
 670 define policy on how often approvals need to be revalidated for each information system, taking into
 671 account the impact level of the system and the level of access granted (e.g., whether the access is to a

672 privileged account, whether execution of arbitrary commands is permitted). Ideally, in a centrally
 673 managed system the revalidation process should be continuously monitored and reflect the immediate
 674 need for individuals and systems to have access to the required resources. To facilitate a holistic
 675 review of access to servers and applications, the review of entitlements granted via authorized key-
 676 based trust relationships should be integrated with existing privileged access management reviews.

677 ■ **Termination:** Terminating access for automated processes when an application is decommissioned or
 678 the automated trust relationship is no longer required by the application. Access for interactive users
 679 should be terminated when a person leaves the organization or changes roles.

680 **5.3 Establish Continuous Monitoring and Audit Processes**

681 The purpose of continuous monitoring is to ensure that the processes for provisioning, life cycle
 682 management, and termination are followed and enforced. It is also important for detecting unauthorized
 683 access that may have been illicitly provisioned by attackers as a backdoor into systems. Unauthorized and
 684 misconfigured SSH user keys should be detected. For example, anyone or anything (including malware)
 685 with access to a superuser account can technically configure new authenticators granting access to any
 686 account.

687 One approach is to periodically discover all configured authorized keys for all user accounts on all
 688 servers, check for each authorized key whether there is a corresponding valid approval (and that the key
 689 has the same access restrictions, such as command restriction, as in the approval), and generate alerts
 690 whenever the configured access does not have a matching approval. In other words, this involves
 691 establishing a baseline and whitelisting that baseline to detect any deviations from the accepted baseline.
 692 Ideally, SSH client configurations are also analyzed, and any configured identity keys are checked against
 693 approvals to ensure that the private key corresponding to each approved authorization is not found from
 694 any unauthorized locations.

695 Log data generated by an information system should be analyzed to 1) detect configured SSH user keys
 696 that are not being used and propose them for removal to eliminate unnecessary access grants (many of
 697 these keys likely belong to users who have left the organization); 2) detect and monitor connections from
 698 remote locations and connections crossing authorization boundaries and ensure that they are properly
 699 approved; 3) detect connections using an authorized key from hosts from which connections have not
 700 been authorized (such connections would typically involve leaked credentials and warrant immediate key
 701 rotation); and 4) identify keys that are used from outside the managed environment as “external keys”
 702 requiring manual rotation. Ideally, such log data analysis integrates with an existing SIEM or other log
 703 data collection system used by the organization.

704
 705 Furthermore, continuous monitoring should ensure that remote access and non-local maintenance
 706 (including privileged access) using SSH user keys is properly audited, and that SSH/SFTP file transfers
 707 using SSH user keys and crossing a security zone boundary are properly audited and content-checked and
 708 are not used for unintended purposes (such as executing commands or transferring files in the wrong
 709 direction).

710
 711 Auditing of SSH user keys serves risk analysis and ensures that the provisioning, life cycle management,
 712 and termination processes as well as continuous monitoring are working properly. A comprehensive audit
 713 of SSH user keys for risk analysis purposes should be performed by all organizations that use SSH or
 714 SFTP protocols (including within file transfer products or for network device or hardware/BIOS-level
 715 configuration). Audits for ensuring the functioning of processes, on the other hand, can use representative
 716 sampling and condition detection tests to gain sufficient confidence that the processes are functioning.
 717

718 **5.4 Inventory and Remediate Existing SSH Servers, Keys, and Trust Relationships**

719 Many organizations have thousands of untracked identity keys, authorized keys, and corresponding trust
 720 relationships granting access across a large number of mission critical systems. Existing legacy keys pose
 721 a substantial security risk and make risk analysis difficult if they are not understood. An inventory of all
 722 existing SSH keys and trust relationships must be created and evaluated against defined policies. The
 723 inventory should include the locations (host and account) of all identity and authorized keys and
 724 authorized key restrictions (e.g., source and command restrictions).

725
 726 Remediation is also necessary for compliance. NIST SP 800-53 requires identifying authorized users and
 727 their access rights, understanding and properly authorizing connections between information systems,
 728 applying the principle of least privilege, and managing authorized keys, among other requirements.

729
 730 An inventory and remediation project typically consists of:

- 731
- 732 ■ Mapping all trust relationships which result from deployed identity and authorized keys and requiring
 733 review and approval from their owners (including description of the purpose of the key or other
 734 justification for the existence of the key). Trust relationships created by public key authentication
 735 must be reviewed and evaluated based on the potential for chains of trust relationships coming in and
 736 out of systems to be used to pivot. Source and command restriction information should be included to
 737 enable evaluation of proper access controls and restrictions. Any keys for which such approval is not
 738 deemed appropriate (such keys may be backdoors and may have been left behind by attackers) should
 739 be removed. Ideally, trust relationship information can be imported into existing privilege access
 740 management systems for review through standard processes.
- 741 ■ Adding command and source restrictions to authorized keys wherever possible
- 742 ■ Identifying and removing any orphan authorized keys for which a corresponding identity key has not
 743 been found, as they may represent backdoor keys which can be used for nefarious purposes
- 744 ■ Identifying and remediating any duplicated identity keys
- 745 ■ Ensuring all identity keys used by interactive users are protected with a passphrase
- 746 ■ Replacing all keys that do not comply with key length and algorithm policies
- 747 ■ Enforcing the SSH identity key provisioning process
- 748 ■ Configuring SSH servers to log fingerprints
- 749 ■ Locking down keys by moving them to root-owned locations (this is needed to produce a stable state
 750 for the later steps)
- 751 ■ Assigning owners (e.g., application, business process owner) for all keys granting access to servers
 752 (e.g., based on the server and account they grant access to, possibly automatically using information
 753 from a suitable configuration database)
- 754 ■ Monitoring log data for several months to determine which keys are not being used, proposing such
 755 keys for removal to their owners, and removing any unused keys that are not specifically approved for
 756 keeping (the monitoring period should usually be 6 to 12 months to ensure all active keys are used;
 757 the monitoring period should include testing of disaster recovery systems and other processes that
 758 may utilize keys)

759 ■ Analyzing which keys grant access across authorization boundaries or other defined boundaries (e.g.,
 760 development/test to production systems, between information systems, remote access) and either
 761 categorically removing authorized keys granting such access or obtaining required approvals for such
 762 access

763 ■ Analyzing which keys are external keys, i.e., used with hosts outside the managed environment

764 ■ Rotating (i.e., changing) all identity keys and corresponding public keys (external keys may need
 765 manual handling and coordination with administrators of the external hosts with which they are used)

766 ■ Onboarding all relevant hosts into a system for managing SSH keys

767 During the remediation project, relevant SSH configurations should be backed up before making a change
 768 so that the change can be quickly rolled back in case something breaks.

769
 770 A key remediation project can be a sizable effort, but it is critical for security and compliance. Note that it
 771 is generally considered infeasible to maintain a 100% inventory of all SSH identity keys because they are
 772 such small files that can be stored anywhere, even on paper (they are small enough to be typed in). The
 773 real goal is to keep an inventory of all authorized keys and to the extent possible, the corresponding
 774 identity keys.

775

776 5.5 Automate Processes

777 The automation of the processes involved in the management of SSH key-based access can significantly
 778 improve security, efficiency, and availability. This includes the following processes:

779

780 ■ **Inventory:** Manual discovery and inventory of all SSH identity and authorized keys (along with
 781 corresponding restrictions) and mapping of all resulting trust relationships is practically impossible.
 782 To accurately complete an SSH key inventory across an enterprise, or even a single department,
 783 automation of some sort is essentially a requirement. Initial identification of all SSH servers to start
 784 the process of key discoveries can be performed by leveraging existing vulnerability scanning
 785 systems.

786 ■ **SSH Key-Based Access Review:** Automatically importing trust relationship data into an existing
 787 privilege access management system for review leverages existing processes and facilitates an
 788 accurate review and tracking of approvals for access granted through SSH keys.

789 ■ **Provisioning:** System administrators in many organizations spend a substantial amount of time
 790 configuring and managing SSH user keys. Many organizations already use a ticketing or change
 791 control system for approving changes to their IT systems. Provisioning interactive or automated
 792 access using SSH is most naturally performed using the same system used for approving other IT
 793 changes or other access. Ideally, a special template is used for filling a request for interactive or
 794 automated access, and once approved, the request is automatically picked up by a key management
 795 system and implemented on all affected hosts. Such automation eliminates the manual steps for
 796 setting up keys, eliminates the need for manual root access for installing the keys, reduces the amount
 797 of privileged administrative access that needs to be audited and reviewed, eliminates configuration
 798 errors due to incorrectly implemented requests, and ensures that the approval template remains
 799 available for future reference and for use in continuous monitoring and audits.

800 ■ **Continuous Monitoring:** Automation is also a virtual necessity for any continuous monitoring
 801 process.

802 ■ **Audit:** A system that can provide central reporting and analysis tools across the inventory can reduce
803 the time necessary to prepare the data required for audits and make it easier for auditors to verify
804 proper implementation and identify exceptions.

805 **5.6 Educate Executive Management**

806 Many executives are not aware of the central role SSH keys play in the secure management and operation
807 of mission critical infrastructure and the significant breaches that can occur if they are exploited. Without
808 sufficient executive education for both security and operationally focused executives, SSH key
809 management initiatives can get derailed by other seemingly higher priorities, leaving an organization
810 vulnerable. To ensure the proper level of priority is placed on the implementing and sustaining SSH key
811 management, it is important that executives understand the systems on which SSH is used, the levels of
812 access granted to users and automated processes via SSH, the risks and potential impacts of an SSH-based
813 breach (including pivoting), and the basic steps needed to implement an SSH key management program
814 (as outlined in this document).
815

816 6. SSH-Based Access Management Planning and Implementation

817 This section discusses considerations for planning and implementing the management of SSH keys for
 818 interactive and automated access. It assumes that the organization's processes involving SSH-based
 819 interactive and automated access management are primarily ad hoc—lack of policies and requirements,
 820 lack of standardized processes and automated tools for key management, etc. As with any technology
 821 deployment, planning and implementation of SSH key management for interactive users and automated
 822 processes should be addressed in a phased approach. A successful deployment can be achieved by
 823 following a clear, step-by-step planning and implementation process. The use of a phased approach for
 824 deployment can minimize unforeseen issues and identify potential pitfalls early in the process. This model
 825 also allows for incorporating advances in new technology and adapting the technology to the ever-
 826 changing enterprise. The following is an example of planning and implementation phases:

- 827 1. **Identify Needs.** The first phase involves identifying the needs to manage SSH keys for interactive
 828 and automated access and determining how those needs can best be met.
- 829 2. **Design the Solution.** The second phase involves all facets of designing the solution. Examples
 830 include cryptographic settings, key management, and management automation.
- 831 3. **Implement and Test a Prototype.** The next phase involves implementing and testing a prototype of
 832 the designed management solution in a lab or test environment. The primary goals of the testing are to
 833 evaluate the functionality, performance, scalability, and security of the management solution, and to
 834 identify any issues with the components, such as interoperability issues.
- 835 4. **Deploy the Solution.** Once the testing is completed and all issues are resolved, the next phase
 836 includes the gradual deployment of the management solution throughout the enterprise.
- 837 5. **Manage the Solution.** After the management solution has been deployed, it is managed throughout
 838 its lifecycle. Management includes maintenance of its components and support for operational issues.
 839 The lifecycle process is repeated when enhancements or significant changes need to be incorporated
 840 into the solution.

841 This document does not describe the planning and implementation process in depth because the same
 842 basic steps are performed for any security management technology. This section only highlights those
 843 considerations that are particular to managing SSH keys for interactive or automated access and were not
 844 already discussed in previous sections of this document.

845 Organizations may follow a project management methodology or life cycle model that does not directly
 846 map to the phases in the model presented here, but the types of tasks in the methodology and their
 847 sequencing are probably similar.

848 6.1 Identify Needs

849 The purpose of this phase is to identify the needs to manage SSH-based interactive and automated access
 850 and determine how those needs can best be met through use of automated enterprise management tools,
 851 ad hoc implementation of a key management system, etc. Requirements specific to identifying needs that
 852 should be considered include the following:

- 853 ■ **Engage Stakeholders.** Because SSH use spans so many technologies and groups, it is important to
 854 identify and engage stakeholders, including groups with large numbers of SSH-based systems (e.g.,

855 Unix and Linux teams, applications, groups leveraging cloud services), auditors, and executive
856 management to understand needs and requirements.

857 ■ **Current Management Processes.** A good understanding of existing SSH key management
858 processes, including any existing automated management tools or scripts is helpful.

859 ■ **Existing Requirements.** There may already be mandates, regulations, organization policies, etc. that
860 prescribe requirements for managing SSH keys used for interactive and automated access.

861 ■ **Volume of Activity.** This should include the approximate number of devices and users/processes
862 currently using SSH-based interactive and automated access, and the approximate number of
863 additional devices and users/processes that could benefit from SSH-based access.

864 ■ **System and Network Environments.** It is important to understand the characteristics of the
865 organization's system and network environments so that a management solution can be selected that
866 will be compatible with them. Aspects to consider include the following:

867 – The characteristics of the devices that act as servers, clients, and management administrators,
868 especially the OSs and SSH application versions (servers or clients) they use and their security
869 postures (low-impact vs. moderate-impact, etc.)

870 – The technical attributes of the interfaces of other systems with which the SSH key management
871 solution might be integrated, such as centralized logging servers, ticketing systems, and security
872 information and event management (SIEM) software

873 The outcome of the analysis is documentation of the requirements for the SSH key management
874 solutions, including security capabilities (e.g., cryptography, key management), performance
875 requirements, management requirements (including reliability, interoperability, scalability), the security
876 of the technology itself, usability (by both administrators and users), and maintenance requirements (such
877 as applying updates). These requirements will be used to design and test the solution.

878 6.2 Design the Solution

879 The next phase is to design an SSH key management solution that meets the identified needs and
880 requirements. If these design decisions are incorrect, then the interactive and automated SSH access
881 implementation will be more susceptible to compromise. Major aspects of solution design that are
882 particularly important for SSH-based access management and have not already been covered in this
883 publication are as follows:

884 ■ **Support for policies and procedures.** The solution must support the implementation, tracking, and
885 enforcement of defined SSH key management policies and procedures. The solution must also
886 support the establishment and enforcement of roles and responsibilities through access controls and
887 workflows.

888 ■ **Cryptographic key management and protection.** Key management and protection is another
889 important component of solution design, including key generation, use, storage, recovery, and
890 destruction.¹⁵ Organizations also need to ensure that access to keys is properly restricted.

¹⁵ NIST SP 800-57, *Recommendation for Key Management*, provides detailed information on key management planning, algorithm selection and appropriate key sizes, cryptographic policy, and cryptographic module selection.

- 891 ■ **Management automation.** The sheer number of SSH keys and complexity of resulting trust
 892 relationships in most organizations makes effective management via manual processes nearly
 893 impossible. As much of the management as possible should be automated, so as to maximize security
 894 and minimize human error and resource expenditures. The areas to target for automation include:
- 895 ○ Discovery and inventory of all identity and authorized keys and corresponding configuration data
 896 (e.g., source and command restrictions). Tracking of “owners” responsible for management and
 897 review of assets within the inventory.
 - 898 ○ Trust relationship mapping and integration with privileged access management systems/processes
 899 for consolidated and regular review of granted access
 - 900 ○ Provisioning and removal of keys, including integration with identity management and other
 901 systems to automatically terminate access (remove authorized keys) on change of role or
 902 termination
 - 903 ○ Continuous monitoring of deployed keys and configuration and validation based on approvals,
 904 including the option to automatically remediate unauthorized changes¹⁶
 - 905 ○ Notification of responsible parties on notable events (e.g., potential rogue keys, unauthorized key
 906 changes, key expiration, weak key lengths, required access reviews)
 - 907 ○ Reporting and dashboard views to enable ongoing analysis and identification of potential
 908 anomalies and vulnerabilities not caught by automated monitoring processes
 - 909 ○ Audit, including automated collection and consolidation of data required for review
- 910 Automating these management functions may involve acquisition of tools and/or development of
 911 scripts. The security of all automation components must be carefully reviewed since, by definition,
 912 they will operate with administrative-level privileges.
- 913 ■ **Integration with existing systems.** The management solution must be able to integrate with existing
 914 systems and infrastructure, including ticketing systems, SIEM systems, configuration management
 915 databases (CMDBs), and privileged access management systems. The solution must address the
 916 management of SSH identity and authorized keys across all technologies where they are used—either
 917 via automated or manual processes.

918 6.3 Implement and Test Prototype

919 After the management solution has been designed, the next step is to implement and test a prototype of
 920 the design. Ideally, implementation and testing should first be performed on lab or test devices. Only
 921 implementations in final testing should be conducted on production devices. Aspects of the solution to
 922 evaluate include the following:

- 923 ■ **Key Management.** Each key should be properly generated, deployed, protected, rotated, and
 924 terminated. Each key should only grant the necessary access to authorized resources for authorized
 925 users/devices.
- 926 ■ **Administration.** Administrators should be able to configure and manage all components of the
 927 solution effectively and securely. It is particularly important to evaluate the ease of deployment and
 928 configuration, including how easily the solution can be managed as the solution is scaled to larger

¹⁶ Note that automatic remediation may inadvertently disable critical SSH-enabled capabilities, so it should be used with caution, if at all.

929 deployments. Another concern is the ability of administrators to disable configuration options so that
930 users cannot circumvent the intended security. Management concerns should include the effects of
931 changing software settings (e.g., changing cryptographic algorithms or key sizes.)

932 ■ **Logging and Auditing.** Logging and auditing for the management solution should function properly
933 in accordance with the organization's policies and strategies.

934 ■ **Security of the Implementation.** The management solution itself may contain vulnerabilities and
935 weaknesses that attackers could exploit. Organizations should perform extensive vulnerability
936 assessments against the management solution components because of their high value. Another
937 common security concern is the security of the cryptographic keys.

938 Organizations should consider implementing the components in a test environment first, instead of a
939 production environment, to reduce the likelihood of implementation problems disrupting the production
940 environment. When the components are being deployed into production, organizations should initially use
941 the management solution for a small number of hosts. Many of the problems that occur are likely to occur
942 on multiple hosts, so it is helpful to identify such problems either during the testing process or when
943 deploying the first hosts, so that those problems can be addressed before widespread deployment. A
944 phased deployment is also helpful in identifying potential problems with scalability.

945 **6.4 Deploy the Solution**

946 Once testing is complete and any issues have been resolved, the next phase of the planning and
947 implementation model involves deploying the solution. A prudent strategy is to gradually migrate devices
948 and users/processes to the new solution. The phased deployment provides administrators an opportunity
949 to evaluate the impact of the solution and resolve issues prior to enterprise-wide deployment. It also
950 provides time for the IT staff (e.g., system administrators, help desk) and users to be trained.

951 Most of the issues that can occur during deployment are the same types of issues that occur during any
952 large IT deployment.

953 **6.5 Manage the Solution**

954 The last phase of the planning and implementation model is the longest lasting. Managing the solution
955 involves operating the deployed solution and maintaining the policies, software, and other solution
956 components. Examples of typical actions are as follows:

957 ■ Testing and applying patches to solution components

958 ■ Including additional types of server and client devices in the access management solution

959 ■ Performing key management duties (e.g., issuing new keys, revoking keys for compromised systems
960 or departing users)

961 ■ Adapting the policies as requirements change. An example is switching to a stronger encryption
962 algorithm or increasing the key size.

963 ■ Monitoring the access management components for operational and security issues

964 ■ Periodically performing testing to verify that access management is functioning properly

- 965 ■ Performing regular vulnerability assessments
- 966 ■ Preparing devices for retirement or disposal. Devices and media that hold private keys should be
967 sanitized or destroyed, unless the keys have been retired/rotated.¹⁷

¹⁷ For more information on media sanitization, see NIST SP 800-88 Revision 1, *Guidelines for Media Sanitization* (<http://csrc.nist.gov/publications/PubsSPs.html#800-88>).

968 **7. Solution Planning and Deployment**

969 Earlier sections have provided an introduction to authentication methods in SSH, with a view towards
 970 automated access, described threats and risks associated with poorly managed SSH keys, and provided
 971 guidelines on how SSH keys should be managed. This section strives to answer the question, how should
 972 one practically proceed to address SSH key management issues?

973 A typical SSH access management or key management project involves:

974 ■ Understanding the current situation, as most organizations have a substantial installed base of SSH
 975 keys that have not previously been managed

976 ■ Procuring and deploying tools and establishing processes for efficient, controlled provisioning of
 977 automated SSH-based access and enforcing approvals according to policy

978 ■ Remediating the existing environment, including establishing purpose for every configured
 979 authorized key, removing any authorized keys and identity keys that are not used or for which no
 980 justifiable purpose can be identified, configuring command restrictions, and rotating any remaining
 981 user keys.

982 An understanding of the existing environment can be obtained through audits. Tools are available that
 983 can help in the audit (see Appendix C for a list of tool selection criteria).

984 Deploying the tools and establishing provisioning processes typically includes locking down keys and
 985 integrating into existing IT change control or approval systems. (The integration may be performed at a
 986 later stage, or in parallel with the remediation phase.)

987 In a typical remediation process, each host is taken under management using the tools, each host is
 988 monitored for a period of time (preferably several months) to identify which keys are actually used and
 989 how, unused/unneeded/rogue keys are removed, trust relationships that cross boundaries inappropriately
 990 or violate policy are removed, command restrictions (and source restrictions, if applicable) are added, and
 991 all keys are rotated. The remediation process is fairly labor-intensive. Depending on how many keys
 992 there are, how readily the owner and purpose of each key can be identified, and how much work is needed
 993 for writing change requests and waiting for maintenance windows, remediating a host may take several
 994 hours of work (tools can help a lot, but they cannot determine the purpose or owner of each key if that
 995 information hasn't been recorded anywhere).

996 Generally it has been found that the cost of manual labor in a key management project is often as big or
 997 even significantly bigger than the cost of the tools for managing the keys. The choice of tools has a major
 998 impact on the amount of manual labor needed.

999 **Appendix A—NIST SP 800-53 Controls Mapping**

1000 This appendix lists the NIST SP 800-53 Revision 4 security controls that are most pertinent for securing
 1001 SSH-based automated access management. Next to each control is an explanation of its implications
 1002 particular to SSH-based automated access management.

1003

NIST SP 800-53 Controls	SSH Implications
AC-2, ACCOUNT MANAGEMENT, control #d	SSH user keys authorize access to the information system and specify privileges for access.
AC-2, ACCOUNT MANAGEMENT, control #g	Enhanced auditing of SSH should be enabled to track the usage of keys and provide an audit trail of which source user (and client) is using keys to connect to the destination user.
AC-2, ACCOUNT MANAGEMENT, control #i	Valid authorization should be required before installing an SSH authorized key, because such keys grant access to the system. Individual users or administrators should not be able to grant access to friends or colleagues without control. Granted key-based access should be limited to intended usage (e.g., intended command).
AC-2, ACCOUNT MANAGEMENT, control #j	SSH keys should be monitored periodically for compliance with key management policies.
AC-2, ACCOUNT MANAGEMENT, control #k	Any private keys held by a group of individuals should be rotated whenever an individual is removed from the group (note: administrators may obtain copies of keys when using service accounts). Keys stored on shared accounts on jump servers should be rotated when someone's access to the jump server is terminated.
AC-3, ACCESS ENFORCEMENT	Approvals for key-based access should be enforced. Users should not be able to install unapproved keys (keys must be locked down).
AC-3, ACCESS ENFORCEMENT, control enhancement #3	Users should be prevented from propagating their access rights by installing new private keys.
AC-4, INFORMATION FLOW ENFORCEMENT	SSH keys should be managed in order to have control over the flow of information between interconnected systems.
AC-4, INFORMATION FLOW ENFORCEMENT, control enhancement #4	Content of encrypted SSH sessions and SFTP file transfers should be content-checked (e.g., run through data loss prevention (DLP) software, antivirus software).
AC-6, LEAST PRIVILEGE	Command restrictions should be configured for SSH authorized keys whenever possible to limit what can be done with the keys. Key-based access to root accounts should be limited to cases where it is necessary to accomplish the task. Only those authorized keys that are necessary for accomplishing the assigned tasks should be configured. In corollary, it is necessary to know what task each key relates to.
AC-6, LEAST PRIVILEGE, control enhancement #2	SSH authorized keys for privileged accounts should only be configured if the task cannot be accomplished using a non-privileged account.

<p>AC-6, LEAST PRIVILEGE, control enhancement #3</p>	<p>SSH authorized keys providing root or other privileged level access should be approved and documented before they are provisioned. Approved authorized keys should specify the command they authorize to be executed.</p> <p>An inventory of approved authorized keys and trust relationships should be maintained, documenting the rationale for each authorized key.</p>
<p>AC-6, LEAST PRIVILEGE, control enhancement #4</p>	<p>Unauthorized users should not be given access to private keys that grant access to privileged accounts.</p>
<p>AC-6, LEAST PRIVILEGE, control enhancement #5</p>	<p>Organizations should ensure that non-organizational users cannot obtain copies of private keys and that any such copies are rendered inoperative by regular key rotation.</p>
<p>AC-6, LEAST PRIVILEGE, control enhancement #7</p>	<p>Privileges granted by SSH keys should be reviewed and adjusted to reflect organizational/business needs.</p> <p>Inappropriate, unneeded, or unused authorized keys should be removed and command and source restrictions added appropriately.</p>
<p>AC-6, LEAST PRIVILEGE, control enhancement #9</p>	<p>Encrypted SSH connections accessing privileged accounts should be audited.</p>
<p>AC-6, LEAST PRIVILEGE, control enhancement #10</p>	<p>Authorized keys files should be locked down to prevent users from adding their own or other public keys to privileged accounts without formal provisioning or approvals.</p>
<p>AC-17, REMOTE ACCESS</p>	<p>Policy should state whether to allow SSH key-based remote access, and if so, state requirements for key rotation and how to prevent copying of private keys (e.g., key must be stored on smartcard). Remote access should be expressly authorized (not provisioned by individual users), which implies locking down keys.</p>
<p>AC-17, REMOTE ACCESS, control enhancement #2</p>	<p>SSH provides such mechanisms. Host key management should be required for preventing man-in-the-middle attacks.</p>
<p>AC-17, REMOTE ACCESS, control enhancement #4</p>	<p>The purpose of each authorized key granting access to root or other privileged accounts should be documented prior to provisioning access and such should be limited to [the organization-defined needs].</p>
<p>AC-20, USE OF EXTERNAL INFORMATION SYSTEMS</p>	<p>SSH user keys can be used for external access. Terms and conditions should address 1) whether and in what direction is key-based access allowed; 2) whether command restrictions are required; 3) whether privileged external access is permitted; 4) what can be done within each external connection (e.g., is port forwarding allowed); and 5) is DLP required for content.</p>
<p>AU-3, CONTENT OF AUDIT RECORDS, control enhancement #1</p>	<p>Enhanced auditing of SSH should be enabled to track the usage of keys and provide an audit trail of which source user (and client) is using keys to connect to the destination user.</p>
<p>CA-2, SECURITY ASSESSMENTS</p>	<p>Many organizations have more SSH user key based authenticators enabling access than they have interactive accounts. Therefore, SSH user key based access should be addressed in the security assessment plan and included in the security assessment.</p>

CA-3, SYSTEM INTERCONNECTIONS	SSH user keys define permanent trust relationships that interconnect information systems (possibly allowing anyone gaining privileged access to one information system to gain unrestricted access to the other information system). SSH key-based trust relationships between information systems must be 1) expressly authorized; 2) documented (including security requirements, such as key rotation); and 3) periodically reviewed.
CA-5, PLAN OF ACTION AND MILESTONES	Since unmanaged SSH user keys can undermine so many other security controls, the action plan should include corrective actions to address SSH user key-based security weaknesses and deficiencies noted during the assessment.
CA-7, CONTINUOUS MONITORING	SSH-based access should be regularly analyzed as part of a continuous monitoring program to detect unauthorized authorized keys configured by users or system administrators (necessary for maintaining and enforcing security authorizations in dynamic environments).
CA-10, INTERNAL SYSTEM CONNECTIONS	Intra-system SSH connections should be properly authorized and documented. Typically such connections are configured using SSH user keys, so authorization and documentation should be required before provisioning authorized keys.
CM-3, CONFIGURATION CHANGE CONTROL	SSH keys and SSH configuration files are security-sensitive configuration information, and their misconfiguration, modification, or unauthorized disclosure may expose servers to unintended access or vulnerabilities. Thus, changes to them should be reviewed, documented, and audited.
CM-5, ACCESS RESTRICTIONS FOR CHANGE	Provisioning and configuring authenticators for automated access to an account should be a controlled, judged decision, balancing the need for access against the risks and must include consideration of the level of access required. Users and system administrators should not be able to modify SSH keys and configuration files without oversight.
CM-6, CONFIGURATION SETTINGS	Standard configurations for sshd_config settings (including AuthorizedKeysFile and PermitRootLogin parameters) should be defined and applied to all SSH servers. SSH key configurations should be documented and approved. Changes should be monitored.
CP-2, CONTINGENCY PLAN	SSH user keys are frequently used in systems that copy data to disaster recovery sites and implement switching operations to use disaster recovery sites. Poorly managed SSH trust relationships (e.g., ones without command restrictions) may be used to spread an attack between sites. The contingency plan should not rely on switching operations to disaster recovery sites if a trust relationship without a command restriction permits access from one site to another, allowing attacks and malware to spread between sites.
IA-2, IDENTIFICATION AND AUTHENTICATION (ORGANIZATIONAL USERS)	<p>SSH identity keys should be associated with an individual user.</p> <p>Policy should prohibit users from sharing private keys, using another user's private key, or copying/moving a private key to another system.</p> <p>Monitoring of key usage should be performed to detect instances where keys are being shared by multiple users, and compromised/shared keys should be rotated.</p>
IA-3, DEVICE IDENTIFICATION AND AUTHENTICATION	Unique SSH host keys should be used for every host running SSH.

<p>IA-5, AUTHENTICATOR MANAGEMENT</p>	<p>SSH user keys (particularly identity keys) are authenticators. SSH identity key creation and corresponding authorized key installation should involve administrative procedures.</p> <p>Lifetime of SSH user keys should be limited. They should be periodically rotated.</p> <p>Policy should prohibit users from sharing private keys, using another user's private key, or copying/moving a private key to another system. Continuous monitoring and regular key rotation should be used to enforce the policy.</p> <p>Shared accounts on jump servers are group/role accounts, and any private keys stored on such accounts (for access from the jump server to end hosts) should be changed when a user's access to the shared account is terminated (effectively, membership in the group of people with access to the account is terminated).</p>
<p>IA-5, AUTHENTICATOR MANAGEMENT, control enhancement #7</p>	<p>Password-based automated access using passwords coded in scripts or binaries should be prohibited.</p>
<p>IA-8, IDENTIFICATION AND AUTHENTICATION (NON-ORGANIZATIONAL USERS)</p>	<p>The same SSH private key (identity key) should not be used for more than one user.</p>
<p>MA-2, CONTROLLED MAINTENANCE</p>	<p>SSH configurations and authorized keys should be checked after maintenance to ensure they are still properly configured and functioning (e.g., as part of continuous monitoring).</p>
<p>PL-2, SYSTEM SECURITY PLAN</p>	<p>Automated access should be considered when defining and enforcing the authorization boundary.</p> <p>A system that can execute commands on another system (e.g., using SSH keys) should be considered to have at least the same security categorization as the system(s) it can access.</p> <p>Trust relations and automated SSH connections with/to other systems should be documented in the security plan.</p>
<p>PL-8, INFORMATION SECURITY ARCHITECTURE</p>	<p>Automated access and SSH key-based access must be considered in the information security architecture. Given that many organizations have more SSH user keys granting access than they have PKI tokens or passwords, they cannot be ignored when developing the information security architecture.</p>
<p>PS-4, PERSONNEL TERMINATION</p>	<p>SSH keys are authenticators that may be associated with an individual. Some individuals may also have had access to authenticators used for shared accounts or privileged accounts (e.g., SSH private keys stored on servers).</p> <p>Any authenticators associated with an individual should be terminated upon the individual's employment - this means at minimum removing the related authorized keys from all servers.</p> <p>Any authenticators that the individual may have had access to should be rotated (i.e., changed) to ensure that copies of the authenticators do not remain operable.</p>
<p>PS-6, ACCESS AGREEMENTS</p>	<p>Access should not be provisioned without proper access agreement. This is one more reason why users or system administrators provisioning key-based access to themselves or others without proper approvals is not acceptable.</p>

<p>RA-3, RISK ASSESSMENT</p>	<p>Risk assessment should consider attack propagation risk: if one system is compromised, an attack could spread to other systems using SSH keys (especially if command restrictions are not used), possibly even to backup systems and disaster recovery sites.</p> <p>The potential impact of an SSH key compromise should be part of the risk assessment process.</p> <p>Authorized keys grant access to an information system. It is impossible to assess the risk facing an information system without knowing and having control of who can access the system and what other systems the system under inspection can access without further authentication.</p>
<p>SC-12, CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT</p>	<p>SSH keys should be rotated and reissued on a regular basis.</p> <p>SSH user keys should be created only through a controlled provisioning process.</p>
<p>SC-23, SESSION AUTHENTICITY</p>	<p>Unique host keys should be used for every SSH server, as that prevents man-in-the-middle attacks.</p>
<p>SI-4, INFORMATION SYSTEM MONITORING</p>	<p>SSH connections using key-based authentication should be monitored and checked against authorized connections to detect unauthorized use of authorized keys (e.g., using copied identity keys) in real time.</p>
<p>SI-7, SOFTWARE, FIRMWARE, AND INFORMATION INTEGRITY</p>	<p>Authorized keys and other SSH configuration files installed without proper authorization should be detected (e.g., using continuous monitoring tools).</p>

1004

1005 **Appendix B—Cybersecurity Framework Subcategory Mapping**

1006 This appendix lists the Cybersecurity Framework Version 1.0 subcategories that are most pertinent for
 1007 securing SSH-based automated access management. Next to each control is an explanation of its
 1008 implications particular to SSH-based automated access management.

1009

Cybersecurity Framework Subcategory	SSH Implications
ID.AM-3: Organizational communication and data flows are mapped	SSH user keys define permanent trust relationships that interconnect information systems (possibly allowing anyone gaining privileged access to one information system to gain unrestricted access to the other information system).
ID.RA-5: Threats, vulnerabilities, likelihoods, and impacts are used to determine risk	<p>Risk assessment should consider attack propagation risk: if one system is compromised, an attack could spread to other systems using SSH keys. The potential impact of an SSH key compromise should be part of the risk assessment process.</p> <p>Authorized keys grant access to an information system. It is impossible to assess the risk facing an information system without knowing and having control of who can access the system and what other systems the system under inspection can access without further authentication.</p>
PR.AC-1: Identities and credentials are managed for authorized devices and users	SSH user keys authorize access to the information system and specify privileges for access. SSH identity keys should be associated with an individual user. Unique SSH host keys should be used for every host running SSH.
PR.AC-3: Remote access is managed	SSH can provide key-based remote access. Remote access should be expressly authorized (not provisioned by individual users). There should be requirements for remote access, such as requirements for key rotation and preventing copying of private keys.
PR.AC-4: Access permissions are managed, incorporating the principles of least privilege and separation of duties	Command restrictions should be configured for SSH authorized keys whenever possible to limit what can be done with the keys. Key-based access to root accounts should be limited to cases where it is necessary to accomplish the task. Only those authorized keys that are necessary for accomplishing the assigned tasks should be configured.
PR.DS-2: Data-in-transit is protected	SSH connections using key-based authentication protect the confidentiality and integrity of data in transit.
PR.DS-6: Integrity checking mechanisms are used to verify software, firmware, and information integrity	Authorized keys and other SSH configuration files installed without proper authorization should be detected (e.g., using continuous monitoring tools).
PR.IP-3: Configuration change control processes are in place	SSH keys and SSH configuration files are security-sensitive configuration information, and their misconfiguration, modification, or unauthorized disclosure may expose servers to unintended access or vulnerabilities. Thus, changes to them should be reviewed, documented, and audited.
PR.PT-1: Audit/log records are determined, documented, implemented, and reviewed in accordance with policy	Enhanced auditing of SSH should be enabled to track the usage of keys and provide an audit trail of which source user (and client) is using keys to connect to the destination user.

Cybersecurity Framework Subcategory	SSH Implications
PR.PT-3: Access to systems and assets is controlled, incorporating the principle of least functionality	SSH user keys define permanent trust relationships that interconnect information systems (possibly allowing anyone gaining privileged access to one information system to gain unrestricted access to the other information system). SSH key-based trust relationships between information systems must be 1) expressly authorized; 2) documented (including security requirements, such as key rotation); and 3) periodically reviewed.
DE.CM-7: Monitoring for unauthorized personnel, connections, devices, and software is performed	SSH-based access should be regularly analyzed as part of a continuous monitoring program to detect unauthorized authorized keys configured by users or system administrators (necessary for maintaining and enforcing security authorizations in dynamic environments).

1010

1011

1012 Appendix C—SSH Key Management Tool Selection

1013 The following areas should be considered when selecting and procuring tools for managing SSH keys:

- 1014 ■ **Enterprise Platform:** Support for the functionality required for enterprise deployments, including:
- 1015 ○ Load balancing and failover so that failure of a single component does not affect system
1016 availability and operation.
 - 1017 ○ Geographic distribution of components for disaster recovery.
 - 1018 ○ Effective operation across firewalls and network boundaries so that all network zones can be
1019 managed.
 - 1020 ○ Secure architecture, including configurable authentication (password, smartcard, multifactor),
1021 granular access controls, and secure logging of all management and operational events.
 - 1022 ○ Third party security assessments and code reviews.
 - 1023 ○ Ability to scale to sustainably manage the current number of SSH assets and planned growth,
1024 including potential acquisitions.
 - 1025 ○ Application programming interfaces (APIs) to programmatically access all relevant functionality
1026 of the system, including configuration, writing and retrieving data, triggering provisioning
1027 operations, discovery, reporting, etc. Support for multiple languages and remote access to APIs.
1028 Permissions must be enforced to restrict functional and data access via APIs to authorized
1029 individuals or processes.
 - 1030 ○ Ability to integrate with enterprise infrastructure and systems, including enterprise directories,
1031 identity management, ticketing, SIEM systems, CMDBs, hardware security modules (HSMs),
1032 workflow, etc.
 - 1033 ○ Ability to optionally manage all cryptographic assets (e.g., certificates and symmetric keys) from
1034 same platform.
- 1035 ■ **Discovery:** Automated discovery of all SSH keys and configuration information, including the ability
1036 to:
- 1037 ○ Perform discovery with or without an agent of all SSH key information, including identity keys,
1038 authorized keys, authorized key restrictions, configuration information, file permissions, and log
1039 information regarding key usage. Discover and identify encrypted identity keys.
 - 1040 ○ Identify systems that have not yet been scanned by analyzing discovered data, such as orphan
1041 keys (authorized keys for which no corresponding identity keys have been found) and logs that
1042 show access to or from other systems on which scanning has not yet been performed.
 - 1043 ○ Optionally network discover SSH servers or import data about SSH servers from other systems
1044 (e.g., vulnerability scanning) that are already performing network scans.
 - 1045 ○ Support discovery of multiple types of systems (e.g., Unix, Linux, Windows, appliances) and
1046 vendor configurations/formats to ensure comprehensive, organization-wide discovery.
 - 1047 ○ Use SSH configuration files to determine the expected location of identity and authorized keys,
1048 and scan those locations or configure scanning of “non-standard” locations.

- 1049 ○ Effectively scan mounted volumes (e.g., Network File System [NFS] or Common Internet File
1050 System [CIFS]) and provide an accurate representation of resulting logical key configuration.
- 1051 ○ Configure scanning schedules uniquely for different groups of assets, providing options for
1052 different intervals (hourly, daily, weekly, monthly, quarterly, annually) and times of day. Tune
1053 scanning parameters to ensure that scan processing does not impact the operation and
1054 performance of applications and processes on production systems.
- 1055 ■ **Inventory:** Cataloging of all SSH assets, including the ability to:
- 1056 ○ Centrally catalog all relevant and discovered information about SSH assets, including keys, key
1057 locations (file paths, server and client addresses), associated account information, and
1058 configuration information (e.g., authorized key restrictions).
- 1059 ○ Optionally manually add keys to the inventory via user interfaces or bulk import to accommodate
1060 situations where automated discovery is not possible.
- 1061 ○ Set policies for critical attributes—such as key length, allowed algorithms, source and command
1062 restrictions, and cryptoperiods—and identify keys that are out of compliance with those policies.
- 1063 ○ Assign asset “owners” and delegate administration of SSH assets through the assignment of
1064 granular permissions and access controls.
- 1065 ○ Assign approvers to SSH assets to ensure that access to each asset has been approved by the
1066 appropriate people.
- 1067 ○ Configure rules for automatically organizing and grouping discovered assets based on attributes
1068 of those assets to facilitate delegation.
- 1069 ○ Add custom metadata fields to SSH assets to enable the collection of data required by internal
1070 processes (e.g., business owner, department, environment, cost center, application ID).
- 1071 ○ Automatically send inventory information to other systems on a scheduled basis (e.g., sending
1072 trust relationship information to a privileged access management system so that SSH-based
1073 entitlements may be reviewed, or updating a CMDB for central tracking of system configuration
1074 information).
- 1075 ■ **Analysis and Reporting:** Visual and programmatic tools for ongoing analysis, review, and potential
1076 identification of anomalies and vulnerabilities not caught by automated monitoring processes,
1077 including:
- 1078 ○ Ability to view, analyze, and report on inventory contents, including sorting, filtering, and export.
- 1079 ○ Identifying potential security issues, including backdoor keys and duplicated or shared private
1080 keys, and boundary crossing (e.g., development to production).
- 1081 ○ Automatically mapping trust relationships for review and analysis. Identification of trust
1082 relationship configuration that may enable pivoting.
- 1083 ○ Dashboard views that provide a high-level view of critical statistics (e.g., deployed key lengths,
1084 algorithms, duplicate keys, potential backdoor keys, assets that are out of compliance) and ability
1085 to drill down to view applicable keys. Customization of dashboard views to allow the most
1086 important information to be shown.

- 1087 ○ Automated generation of reports that provide detailed information on a variety of critical statistics
- 1088 (e.g., key lengths, algorithms, old keys, trust relationships):
- 1089 ▪ Ability to customize contents and formatting of reports.
- 1090 ▪ Ability to set delivery schedules for reports and configure intended recipients for reports.
- 1091 ▪ Ability to configure individualized reports to include only the responsible group or
- 1092 individual's assets within the reports.
- 1093 ■ **Provisioning:** Adding, configuring, removing, and rotating SSH keys, including the ability to:
- 1094 ○ Set one or more policies to require specific configuration options during provisioning and
- 1095 management operations (e.g., key lengths, algorithms, cryptoperiods, source and command
- 1096 restrictions, environment crossing).
- 1097 ○ Manage lifecycle of identity and authorized keys, including generation of identity keys, adding
- 1098 and removing authorized keys, and integration with identity management and other systems to
- 1099 automatically terminate access (remove authorized keys) on change of role or termination.
- 1100 ○ Automatically rotate identity and authorized keys prior to end of configured cryptoperiods.
- 1101 ○ Support configuration and management of encrypted identity keys.
- 1102 ○ Rollback key rotations in case an issue is encountered with new keys.
- 1103 ○ Designate external keys (e.g., an authorized key that is configured to allow access from an
- 1104 identity key held by another organization).
- 1105 ○ Restrict permissions on SSH key and configuration files (e.g., authorized keys) or relocate
- 1106 authorized keys to a secure location to prevent unauthorized changes.
- 1107 ○ Provide a user portal or other automated mechanism for requesting new identity keys or new
- 1108 access (via authorized keys).
- 1109 ○ Configure and require approvals (dual control) for key management operations, including key
- 1110 generation, adding/removing authorized keys, rotations, etc. Log approvals and associated
- 1111 information. Allow for assignment of one or more approvers or groups of approvers.
- 1112 ○ Automatically replace large numbers of SSH keys (e.g., in case of a broad system compromise).
- 1113 ○ Support deployed SSH products for provisioning and management operations.
- 1114 ○ Integrate with ticketing and change control systems to trigger, approve, and track provisioning
- 1115 operations.
- 1116 ■ **Continuous Monitoring:** Provide continuous monitoring of SSH keys, including:
- 1117 ○ Verification of deployed configuration and keys against approved configuration.
- 1118 ○ Configurable notifications when unauthorized keys or configurations are detected.
- 1119 ○ Option to automatically remediate out-of-compliance situations (e.g., rollback changed keys,
- 1120 deleted authorized keys).
- 1121 ○ Track and monitor key usage and identify unused keys.

- 1122 ○ Ensure regular reviews of SSH-based entitlements are performed and reapproved.
- 1123 ○ Identify keys and configuration that are out of compliance with configured policies.
- 1124 ○ Tuning and scheduling frequency of monitoring processes.
- 1125 ■ **Notifications and Alerting:** Ability to automatically send notifications/alerts based on specific
- 1126 events, including the ability to:
 - 1127 ○ Configure which events for which to send notifications (e.g., detected unauthorized keys, errors,
 - 1128 keys older than cryptoperiods) and set thresholds where applicable.
 - 1129 ○ Configure different target recipients for specific events and groups of assets.
 - 1130 ○ Escalate alerts if action is not taken within a configurable period of time.
 - 1131 ○ Configure methods of notification, including email, Simple Network Management Protocol
 - 1132 (SNMP), syslog, or direct integration with SIEM, ticketing, or other systems.
- 1133 ■ **Auditing:** Ability for auditors to perform reviews of the overall SSH environment and SSH
- 1134 management system, including:
 - 1135 ○ Logging of all relevant operations and management actions within the management solution to
 - 1136 enable forensic-level analysis.
 - 1137 ○ Ability to view, sort, and filter log events.
 - 1138 ○ Read-only access to reports and dashboard functionality to enable high-level review and
 - 1139 drilldown on all relevant aspects of the SSH environment.
- 1140 Given that labor costs of the remediation project and the cost of ongoing key provisioning form the bulk
- 1141 of total costs around managing automated SSH-based access, it is important to consider these aspects of a
- 1142 project and select a product that minimizes the total cost of the project—this may not necessarily be the
- 1143 lowest cost product.
- 1144 A well-designed product can produce operational cost savings and pay for itself by reducing manual labor
- 1145 previously expended on SSH key provisioning, termination, and rotation—and improve operational
- 1146 flexibility by reducing lead times in provisioning—in addition to meeting the security requirements
- 1147 driving the project.
- 1148 If the organization does not have sufficient understanding of SSH keys, automated access management,
- 1149 and the remediation process in-house, use of outside expertise to assist in the process may be warranted.
- 1150
- 1151

1152 **Appendix D—Building Block**

1153 This appendix contains a detailed discussion of a particular problem that is relevant across a variety of
1154 industry sectors. The discussion essentially constitutes a “building block” for addressing the challenge.

1155 **D.1 Description**

1156 Interactive and automated access between systems has become increasingly prevalent. Examples of this
1157 access include file transfers, disaster recovery, privileged access management, software and patch
1158 management, and dynamic cloud provisioning. Automated access is often accomplished using SSH;
1159 however, these architectures have been developed on an ad hoc basis, without formal lifecycle
1160 management. This has caused a variety of security issues that may leave SSH-enabled systems more
1161 susceptible to compromise and may make these systems less well-audited, therefore reducing
1162 accountability.

1163 A possible solution for improving the security of SSH-enabled systems that allow interactive and
1164 automated access is to implement management, operational, and technical security control processes that
1165 address the issuance and maintenance of new SSH user keys. This includes auditing the existing user keys
1166 to identify key replacement needs, privilege issues, and other problems. The primary drawback with this
1167 approach is the amount of effort that it may involve, and the resulting expenses that an organization may
1168 have to pay. Imagine having to audit hundreds of thousands of existing user keys within a large
1169 organization. So the goal is to establish an automated access management capability that improves
1170 security without overwhelming the resources of the organization.

1171 **D.2 Scenario**

1172 **Example Scenario – Existing SSH User Keys**

1173 An organization has used SSH user keys to enable system-to-system access for years. There was no
1174 record keeping for these keys and no other lifecycle management activities; keys were simply issued as
1175 requested. To get the situation under control, the organization first implements new policies and
1176 procedures for handling all user keys in the future, including issuing, replacing, and revoking them. Next,
1177 the organization audits the existing server configurations and keys to determine what is needed. Because
1178 the new processes are in place, the old keys can be handled under these processes—such as issuing new
1179 keys and revoking old keys. Once all old keys have been revoked (after being replaced with new keys or
1180 determined to be unnecessary), then the new keys can all be managed under the new lifecycle
1181 management structure.

1182 **D.3 Security Characteristics**

- 1183 ■ Only the necessary SSH user keys are issued. Keys are revoked when no longer needed. Keys that are
1184 needed over a period of time are rotated on a regular basis during that time.
- 1185 ■ Each SSH user key has only the necessary privileges/functionality.
- 1186 ■ Detect suspicious activity more quickly so that inadvertent or intentional damage to organization data
1187 and resources can be reduced

1188 **D.4 Approach**

1189 This building block is intended to demonstrate security capabilities that can better safeguard automated
1190 access between systems. As described throughout this publication, implementing a variety of

1191 management, operational, and technical security controls is proposed as the fundamental approach to
 1192 solving this problem.

1193 One of the most important parts of this solution is the audit of existing SSH user keys. This audit is
 1194 critically necessary for determining which SSH user keys are necessary and restricting each key to only
 1195 the required privileges/functions. It is important to note that the audit is strictly an inventory and
 1196 assessment exercise; it does not directly involve actions with keys, such as reissuing or revoking them.

1197 Another important facet of this solution is ensuring that future activities involving SSH user keys are
 1198 controlled and monitored. Formal processes, not ad hoc actions, need to be followed to ensure that only
 1199 the necessary keys are issued, that the keys are secured, that the keys are only used as designated, and that
 1200 keys are reissued, revoked, etc. as necessary. If these formal processes are not put into place and
 1201 maintained themselves, security conditions will tend to rapidly deteriorate and the organization will soon
 1202 be in a position where another full-fledged enterprise-wide user key audit will be needed.

1203 **D.5 Business Value**

- 1204 ■ Enable rapid secure deployment of automated system-to-system access for data exchange purposes
- 1205 ■ Prevent unauthorized usage of old SSH user keys, thus preventing potential compromises of
 1206 organization data and resources
- 1207 ■ Prevents unnecessary and unauthorized data accesses/transfers, which could compromise sensitive
 1208 organization data

1209 **D.6 Relevant Standards**

- 1210 ■ Ylonen, T., Kent, G., and Souppaya, M., “[Managing SSH Keys for Automated Access – Current](https://tools.ietf.org/html/draft-ylonen-sshkeybcp-01)
 1211 [Recommended Practice](https://tools.ietf.org/html/draft-ylonen-sshkeybcp-01)”, Internet-Draft, April 2013. [https://tools.ietf.org/html/draft-ylonen-](https://tools.ietf.org/html/draft-ylonen-sshkeybcp-01)
 1212 [sshkeybcp-01](https://tools.ietf.org/html/draft-ylonen-sshkeybcp-01)

1213

1214 **Appendix E—Acronyms and Abbreviations**

1215 Selected acronyms and abbreviations used in this publication are defined below.

AD	Active Directory
AES	Advanced Encryption Standard
BIOS	Basic Input/Output System
CCM	Cipher Block Chaining-Message Authentication Code
CIFS	Common Internet File System
CMAC	Cipher-Based Message Authentication Code
DLP	Data Loss Prevention
DSA	Digital Signature Algorithm
FIPS	Federal Information Processing Standard
FISMA	Federal Information Security Management Act
HMAC	Hash Message Authentication Code
HSM	Hardware Security Module
IT	Information Technology
ITL	Information Technology Laboratory
LDAP	Lightweight Directory Access Protocol
MAC	Message Authentication Code
NFS	Network File System
NIST	National Institute of Standards and Technology
OMB	Office of Management and Budget
OS	Operating System
PIV	Personal Identity Verification
RFC	Request for Comment
SFTP	Secure File Transfer Protocol
SHA	Secure Hash Algorithm
SIEM	Security Information and Event Management
SP	Special Publication
SSH	Secure Shell
TGT	Ticket Granting Ticket
VPN	Virtual Private Network

1216

1217 **Appendix F—Glossary**

1218 Selected terms used in this publication are defined below.

Authorized Key	A public key that has been configured as authorizing access to an account by anyone capable of using the corresponding private key (identity key) in the SSH protocol. An authorized key may be configured with certain restrictions, most notably a forced command and a source restriction.
Authorized Keys File	The file associated with a specific account where one or more authorized keys and optional restrictions are stored. Each account for which public key authentication is allow on an SSH server has a unique authorized keys file.
Automated Access	Access to a computer by an automated process without an interactive user, generally machine-to-machine access. Automated access is often triggered from scripts or schedulers, e.g., by executing an SSH client or a file transfer application. Many programs may also use automated access using SSH internally, including many privileged access management systems and systems management tools.
Automated Process	An application, script, or management system that leverages SSH to execute commands or transfer data to/from another system.
External Key	An authorized key that is used from outside the organization (or outside the environment considered for SSH user key management purposes), or an identity key that is used for authenticating to outside the organization (or outside the environment considered for SSH user key management purposes). Key rotation can break external keys, and therefore it must be ensured that the other side of trust relationships involving external keys is also properly updated as part of rotation. Alternatively, rotation of external keys may be prevented, but that is not a sustainable solution long-term.
Fingerprint	A hash value of a (public) key encoded into a string (e.g., into hexadecimal). Several fingerprint formats are in use by different SSH implementations.
Forced Command	A restriction configured for an authorized key that prevents executing commands other than the specified command when logging in using the key. In some SSH implementations, forced command can be configured by using a "command=" restriction in an authorized keys file.
Host Key	A public key used for authenticating a host in the SSH protocol to hosts that want to communicate with it (each host also generally has its own private host key). Some hosts may have more than one host key (e.g., one for each algorithm). Host keys are used for authenticating hosts (machines) themselves, not users or accounts, whereas identity keys and authorized keys relate to authenticating users/accounts and authorizing access to accounts on hosts.
Identity Key	A private key that is used for authentication in the SSH protocol; grants access to the accounts for which the corresponding public key has been configured as an authorized key.

Interactive User	A person who uses an SSH client to access one or more SSH servers, typically to perform administrative operations, transfer data, or access applications.
Key	A cryptographic key. In this document, keys generally refer to public key cryptography key pairs used for authentication of users and/or machines (using digital signatures). Examples include identity key and authorized keys. The SSH protocol also uses host keys that are used for authenticating SSH servers to SSH clients connecting them.
Key Rotation	Changing the key, i.e., replacing it by a new key. The places that use the key or keys derived from it (e.g., authorized keys derived from an identity key, legitimate copies of the identity key, or certificates granted for a key) typically need to be correspondingly updated. With SSH user keys, it means replacing an identity key by a newly generated key and updating authorized keys correspondingly.
Known Hosts File	A file associated with a specific account that contains one or more host keys. Each host key is associated with an SSH server address (IP or hostname) so that the server can be authenticated when a connection is initiated. The user or administrator who makes the first connection to an SSH server is responsible for verifying that the host key presented by that server is the actual key (not a rogue key) before it gets placed in the known hosts file.
Passphrase	A password used to protect an identity key. After entered by a user or administrator, a passphrase is mathematically converted into large number which serves as a key that is used to encrypt the identity key. In order to decrypt the identity, the passphrase must be entered again so that the same key can be regenerated for decryption.
Pivot	The act of an attacker moving from one compromised system to one or more other systems within the same or other organizations. Pivoting is fundamental to the success of advanced persistent threat (APT) attacks. SSH trust relationships may more readily allow an attacker to pivot.
Public and Private Key	Public and private keys are two very large numbers that (through advanced mathematics) have a unique relationship, whereby information encrypted with one number (key) can only be decrypted with the other number (key) and vice versa. In order to leverage this characteristic for security operations, once two numbers are mathematically selected (generated), one is kept secret (private key) and the other is shared (public key). The holder of the private key can then authenticate themselves to another party who has the public key. Alternatively, a public key may be used by one party to send a confidential message to the holder of the corresponding private key. With SSH, the identity key is a private key and authorized keys are public keys.
Source Restriction	A restriction configured for an authorized key that limits the IP addresses or host names from which login using the key may take place. In some SSH implementations, source restrictions can be configured by using a "from=" restriction in an authorized keys file.

SSH Client	The software implementation that enables a user or an automated process to remotely access an SSH server. An SSH client is responsible for reliably performing all of the operations necessary to ensure a secure connection, including generating identity keys, prompting users to verify host keys, authenticating and establishing encrypted connections with SSH servers, prompting users for credentials, performing public key authentication, etc.
SSH Key	A term that is generally used to refer to an identity and authorized keys. The term may also be occasionally used to refer to host or server private keys.
SSH Server	A software implementation that enables SSH access to a system from SSH clients. SSH server may be included with an operating system or appliance or may be add-on software. An SSH server is typically a complex set of software modules responsible for a broad number of tasks, including enforcing configured SSH settings, authenticating users, limiting access to certain users and groups, ensuring secure connections, interfacing with other systems (e.g., PAM and Kerberos), performing file transfers, etc.
Trust Relationship	The access relationship that is granted by an authorized key in an account on one system (server) and a corresponding identity key in an account on another system (client). Once deployed, these two keys establish a persistent trust relationship between the two accounts/systems that enables ongoing access.
User Key	A key that is used for granting access to a user account via the SSH protocol (as opposed to a host key, which does not grant access to anything but serves to authenticate a host). Both authorized keys and identity keys are user keys. A user key is the equivalent of an access token.

1220 Appendix G—References

1221 References for the publication are listed below.

1222

1223 [ID-SSH] Ylonen, T., Kent, G., and Souppaya, M., “[Managing SSH Keys for Automated Access –](#)
1224 [Current Recommended Practice](#)”, Internet-Draft, April 2013.

1225 [RFC4251] Ylonen, T. and Lonvick, C., “[The Secure Shell \(SSH\) Protocol Architecture](#)”, RFC 4251,
1226 January 2006.

1227 [RFC4252] Ylonen, T. and Lonvick, C., “[The Secure Shell \(SSH\) Authentication Protocol](#)”, RFC
1228 4252, January 2006.

1229 [RFC4253] Ylonen, T. and Lonvick, C., “[The Secure Shell \(SSH\) Transport Layer Protocol](#)”, RFC
1230 4253, January 2006.

1231 [RFC4254] Ylonen, T. and Lonvick, C., “[The Secure Shell \(SSH\) Connection Protocol](#)”, RFC 4254,
1232 January 2006.