ITL BULLETIN FOR JANUARY 2016

SECURING INTERACTIVE AND AUTOMATED ACCESS MANAGEMENT USING SECURE SHELL (SSH)

Murugiah Souppaya, Karen Scarfone,[1] and Larry Feldman,[2] Editors
Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
U.S. Department of Commerce

Introduction

ITL has released an important new guidance document on access management: NIST Internal Report (NISTIR) 7966, *Security of Interactive and Automated Access Management Using Secure Shell (SSH)*. The SSH protocol provides a way to authenticate the identity of users and hosts before allowing them to execute commands on other hosts in either an interactive or an automated fashion. This access is necessary for many purposes, including file transfers, disaster recovery, privileged access management, software and patch management, and dynamic cloud provisioning. Unfortunately, the security of SSH key-based access is often overlooked by organizations, and misuse or compromise of SSH keys could lead to unauthorized access, often with high privileges. Therefore, organizations need to improve their management of SSH user keys, including key provisioning, termination, and monitoring. This publication provides the basics of SSH interactive and automated access management, focusing on explaining how organizations should manage their SSH keys.

SSH Client Authentication Methods

SSH client authentication refers to the authentication of interactive users (administrators and other human users) and automated processes operating through SSH clients. Each user or process authenticates to a particular account on the host running the SSH server. The SSH protocol supports several methods for client authentication, including passwords, host-based authentication, Kerberos, and public key authentication, and one or more of these methods can be enabled on each SSH server. NISTIR 7966 discusses each method in detail, describing the pros and cons of each in terms of security and flexibility. Organizations should carefully evaluate and select the client authentication method or methods that are acceptable for use, and disable the use of all other methods.

NISTIR 7966 recommends the use of public key authentication for automated processes. Public key authentication uses SSH user keys or certificates, typically user keys, to authenticate a connection. This authentication method offers a combination of security features that the other methods do not provide, making it particularly well suited for automated processes. Examples of these features include command

[1] Karen Scarfone is a Guest Researcher from Scarfone Cybersecurity.
[2] Larry Feldman is a Guest Researcher from G2, Inc.

restrictions, which limit what can be done on the server, and source restrictions, which limit which Internet Protocol addresses can establish connections with the server.

Public key authentication is also recommended for interactive users, with smartcard-based solutions being preferred because of their superior security characteristics. The smartcard is used to store and protect the user's identity key. An alternative is to keep the identity key in a password-protected file on the client device. The password is used to decrypt the key, so the strength of the password has a major impact on the security of the key and the SSH access to other hosts that it enables.

Vulnerabilities in SSH-Based Access

SSH is widely used to manage servers, routers, firewalls, security appliances, and other devices through accounts with elevated privileges. This makes SSH keys a particularly attractive target for attackers. Unfortunately, many organizations are not aware of the vulnerabilities inherent in SSH use if proper provisioning, termination, and monitoring processes are not performed, especially when the SSH use includes automated access. NISTIR 7966 describes seven major categories of vulnerabilities:

- **Vulnerable SSH implementation.** The SSH client or server implementation could have exploitable vulnerabilities, including software flaws, configuration weaknesses, and SSH protocol weaknesses.

- **Improperly configured access controls.** The SSH software or components that the SSH software integrates with may not be configured correctly, which could allow unauthorized access to privileged accounts, unauthorized elevation of privileges for standard accounts, and other unintended access.

- **Stolen, leaked, derived, and unterminated keys.** Anyone who has acquired access to an SSH identity key, such as by having malware harvest keys from an organization's laptops or using an old key that should have been terminated, may be able to use that key to gain unauthorized access to one or more of an organization's systems.

- **Backdoor keys.** Organizations often mandate use of a privileged access management system for all privileged access to their servers. However, SSH public key authentication can be used to create a "backdoor." It can be done by generating a new key pair and adding a new authorized key to an authorized keys file that circumvents the privileged access management system and its monitoring and auditing capabilities.

- **Unintended usage.** Users may use SSH identity keys for unintended purposes, such as tunneling traffic instead of performing automated file transfers. This usage—intentional or unintentional—could cause activity to be hidden from network security controls.

- **Pivoting.** Pivoting is the process of an attacker traversing an organization's systems by repeatedly moving from one server to another, often using credentials acquired from servers

along the way. When automated SSH access is allowed, malware on a client system may be able to steal an SSH key and use it to gain access to a server, where it steals more keys and uses them to gain access to other servers.

- **Lack of knowledge and human errors.** SSH management is complex, making it more prone to errors, and many administrators have insufficient knowledge of secure SSH configuration and management practices. A single mistake could provide privileged access to unauthorized users and go undetected for years.

Recommended Practices for Securing SSH Access

Effectively securing SSH access consists of defining clear policies and procedures, and implementing management, operational, and technical security control processes supporting these policies and procedures. The organization should address not only the security of already-deployed SSH systems, privileges, and user keys, but also the security of new SSH systems and user keys.

Examples of practices that should be addressed in an organization's policies and procedures include the following:

- Only enable SSH server functionality on systems where it is absolutely required;
- Keep SSH server and client implementations fully up to date on all systems;
- Harden all SSH server and client implementations;
- Enforce least privileged access for all SSH-accessible accounts;
- Ensure that all SSH user keys (identity and authorized keys) meet minimum requirements, including the following:
    - Use of an approved algorithm and sufficiently long key with an acceptable maximum cryptoperiod (lifetime);
    - Access controls for both identity and authorized keys; and
    - Specification of command and source restrictions for authorized keys used for automated processes.

Provisioning and configuring SSH access to an account should balance the need for access against the risks and should include consideration of the level of access required. Organizations should follow a controlled provisioning and life cycle process. The initial phases of this process are: the Request phase, where someone submits a formal request for establishing SSH access; the Approval phase, where change control processes are used to review and approve or deny the request; and the Provisioning phase, where the approved request is implemented by deploying SSH software and generating and deploying keys. Once the keys are available for use, there is an extended Usage Logging phase, during which all use of the keys is recorded in logs for continuous monitoring, auditing, and forensic purposes.

Periodically, the organization should review and reauthorize each instance of SSH key-based access. When a system or application is decommissioned, an application no longer needs to be administered

remotely, or another change occurs that eliminates the need for SSH usage, the corresponding SSH access should be terminated.

Remediation and Automation

Remediating weaknesses in existing SSH implementations and keys can be a daunting task. Many organizations have thousands of untracked SSH keys granting access across a large number of mission-critical systems. Existing legacy keys pose a substantial security risk. An inventory of the location of all existing keys and an inventory of trust relationships involving these keys should be created and evaluated against defined policies. All issues should be corrected over time through key replacement/rotation or termination, command and source restriction implementation, mandatory identity key authentication, and other means.

Remediation of existing SSH weaknesses and prevention of new SSH weaknesses can both be significantly improved by automating processes. For example, manually discovering and inventorying all SSH identity and authorized keys, then mapping all the trust relationships, is practically impossible; automation is essentially a requirement. The use of automation is also strongly recommended for provisioning purposes, where a single request could affect keys on thousands of hosts. Automation for provisioning eliminates manual steps, reduces privileged administrative access, reduces or eliminates configuration errors, and tracks all changes for use in future audits and in continuous monitoring.

Conclusion

NISTIR 7966 explains the vulnerabilities associated with poor management of interactive and automated SSH access, as well as the potential impact of misuse or compromise of SSH keys used for client authentication. SSH access management is often ad hoc, lacking policies and requirements, and lacking standardized processes and automated tools. Planning and implementing sound management of SSH keys should be addressed in a phased approach following a clear step-by-step process. An example of the phases is identifying needs, designing the solution, implementing and testing a prototype, deploying the solution, and managing the solution. SSH key management should be as automated as possible.

Managing the solution involves general security activities, such as maintaining and enforcing the policies, testing and applying patches, performing continuous monitoring to identify operational and security issues, and conducting regular vulnerability assessments. It also involves several activities particular to SSH access, including performing SSH key management duties and adapting the SSH policies as requirements change (such as switching to a stronger encryption algorithm or a longer minimum key size). Organizations that acquire and use automated SSH key management products should be able to significantly decrease their risks related to SSH access with a reasonable amount of effort. Without automation, most organizations will struggle to remediate the existing SSH environment and to properly secure new SSH usage.

Finally, NISTIR 7966 provides the following lists to assist organizations in implementing SSH security measures:

- NIST Special Publication (SP) 800-53 Revision 4 security controls that are most pertinent for securing SSH-based interactive and automated access management;
- Selected Cybersecurity Framework subcategories with their implications to SSH-based interactive and automated access management; and
- Criteria for selecting SSH key management tools.

ITL Bulletin Publisher: Elizabeth B. Lennon
Information Technology Laboratory
National Institute of Standards and Technology
elizabeth.lennon@nist.gov