Best Practices for Securing Privileged Identities

A Maturity Model

ca technologies

Understanding Your Risk

My organization:

- ☐ Identifies and keeps an inventory of all privileged accounts
- ☐ Has a process to remove all unnecessary privileged access and accounts
- ☐ Controls access to all privileged accounts and technically prevents password sharing
- ☐ Monitors all privileged accounts with video records and can trace all actions back to an individual, even for shared accounts Controls what individual users can do with a privileged account, even if it is shared
- ☐ Protects key server resources from even administrators, including directory controls, application jailing and binary integrity controls
- ☐ Centrally manages all privileged identities in a single user store to reduce administrative errors and gain consistency across platforms
- ☐ Governs privileged identities, including automated provisioning, de-provisioning and entitlement certification

Introduction to Privileged Accounts

Before securing privileged identities, an organization needs to understand the types of accounts that exist as well as their unique purposes and requirements.

Privileged Account Type	Description	Used By	Security Focus
Default Local Administrative Accounts (e.g., root, administrator)	"All powerful" accounts used to manage the system or device by administrators	Multiple types of administrators, often shared	Ensure accountability by individuals.
Named Administrator Accounts	Accounts for individuals that have administrative privileges	Named individual administrators	Ensure least privilege access.
Service Accounts	Accounts used by operating system services and applications that require privileged access, often used by web servers, e-mail servers, databases, etc.	System services and applications	Ensure unused accounts are removed or disabled and that passwords are changed regularly.
Domain Administrator Accounts	Accounts used to administer a domain instead of a local system	Domain administrators	Apply extra controls and monitoring.
Emergency Accounts	Accounts used only in the event of an emergency that requires temporary privileged access	Backup administrators	Ensure all emergency use is authorized and monitor all use.
Application Administrator Accounts	Application accounts with elevated privileges that are used to administer an application	Application administrators	Apply risk-appropriate controls, dependent on the nature of the application.
Hypervisor Administrator Accounts	Accounts used to administer virtual environments, such as VMware	Virtualization administrators	Applying best-practice controls to this class of administrator.

Baseline Protection: Relying on the Operating System

Many organizations apply only minimal controls over privileged accounts, such as restricting access to passwords. This not only leads to a lack of accountability when shared accounts are used but also violates the principle of "least privilege" where all individuals should only be granted the minimum privileges they require.

In other cases, organizations focus on security provided by the operating system, such as group memberships, setuid, and the use of "sudo" on UNIX®/Linux® and "runas" (run as) on Windows®. However, operating system security has critical flaws that make it fundamentally ill-suited to be a primary security tool:

1. **Lack of self-protection.** Operating systems are inherently incapable of ensuring the integrity of their own controls. All systems have privileged accounts which can change or bypass that system's security controls. A user with the proper access can disable the controls necessary to perform an unauthorized action and can modify system log files to erase records of that activity.
2. **Known controls.** Operating system access controls are at risk of being analyzed and avoided because they are known controls. Unauthorized and malicious users often have a deep understanding of how UNIX/Linux and Windows security works. They can use this knowledge to view operating system permissions to look for vulnerabilities. Malicious users will also look to modify operating system logs in order to "hide their tracks." Even on systems where access controls are rigorously enforced, well-trained attackers will simply avoid taking actions that would generate standard alerts that may lead to detection. Only a fully externalized security system can bring unexpected and unknowable elements to a security system and provide the access controls and user activity logs needed to truly secure a system.
3. **Lack of uniformity.** There can be significant variance of capabilities and availability of security controls across platforms (e.g., UNIX file/directory controls are significantly different than Windows). This can lead to tangible security issues such as: Security policies must accommodate systems limitations and do not meet business needs ,Errors and omissions caused by added complexity of security management
4. **Manual Administration.** Many administrative tasks are often performed manually, such as changing passwords and maintaining sudoers files (on UNIX/Linux). When security tasks are performed manually, there is a greater chance of errors and a lack of consistent view and updates.

A Privileged Identity Management Maturity Model

Effectively securing privileged identities requires much more than basic operating security, but security controls must always be tailored to the specific environment and risk-level. It is difficult to implement all possible privileged account controls at the same time. A maturity model is often useful for an organization to understand how they can move from baseline controls to highly-effective privileged account security. CA recommends thinking about implementing privileged identity management in three areas. They may be thought of as sequential steps, although real-world implementation should be tailed to the specific organization:

1. Control access to your privileged accounts.
2. Monitor your privileged accounts.
3. Control what users and applications can do with privileged accounts.
4. Apply complementary controls.

Step 1: Control access to your privileged accounts.

1. **Identify and clean up unused privileged accounts (eliminate identity sprawl).**
   The first step to securing privileged identities is to inventory the privileged accounts in your environment. All accounts with privileges beyond those of a standard user should be recorded, categorized, and understood. All unused and unnecessary accounts should be removed.
2. **Establish a process for granting and removing privileged access.**
   When employees change roles, they often experience what is called "privilege creep" when they gain new access rights without having the access permissions from the old role removed. In addition, individuals who leave an organization are much more likely to abuse their privileges. All privileges should be granted for only as long as required.
3. **Ensure privileged passwords are stored in a secure location.**
   Privileged account passwords should never be stored on sticky notes or locally on an individual's computer. All passwords should be stored in a central system that protects the password from unauthorized access.
4. **Prevent password sharing by making privileged account access seamless.**
   Traditional password management tools allow users to "check out" and "check in" passwords. This exposes the account passwords to being shared with unauthorized users. If an individual is granted access to a privileged identity, they should be able to log on to that account automatically without being provided the account password.
5. **Ensure all passwords to privileged accounts are changed regularly.**
   While individuals are commonly promoted to change their passwords, it is common for shared account and service account passwords to go unchanged for extended periods of time. All passwords–particularly to privileged accounts–need to be changed regularly. Passwords should be automatically changed on a 30-day cycle at the longest. Eliminate any accounts that have passwords that do not expire or set an expiration.
6. **Eliminate all hard-coded passwords for scripts and applications.**
   Hard-coded passwords both prevent proper password updates and also make an easy target for hackers. Applications and scripts should be granted privileges without having to use static passwords.

7. **Require advanced authentication in order to log into all privileged accounts.**
   Passwords alone are not strong enough to be the only barrier to accessing a privileged account. Require one-time passwords, combined with risk-aware authentication to access privileged accounts. Advanced authentication should be required for accessing: 1. administrative accounts on all endpoints, 2. the password safe, 3. Domain administrator accounts, and 4. the privileged identity management administrative console

8. **Remove privileged access from standard user accounts.**
   All administrative tasks should be performed using an account dedicated to privileged actions. Some standard user accounts have been granted administrative privileges. This causes three problems: administrative actions using standard user accounts are difficult to track, using standard user accounts for administrative actions leads to more casual use of privileges which can result in damaging errors and standard user accounts are easier to compromise as they are also often used for web browsing, opening emails and other tasks.

9. **Implement a request workflow for credential access approval including dual-controls and integration with helpdesk ticketing systems.**
   Make access to privileged accounts seamless from a help desk ticketing system. A user who needs to gain access to a privileged account should be granted the required access directly from the ticket without having to go to a separate system.

10. **Eliminate the option of interactive (human) login for service accounts.**
    Service account access should be restricted to operating system services and applications. Remove the ability for a user to log in to these accounts directly.

Step 2: Monitor your privileged accounts.

1. **Keep a video record of all sessions.**
   All sessions should be recorded using a DVR-like tool. Video records should be playable for specific time periods, searchable and track individual actions when using shared accounts

2. **Enable accountability for shared accounts.**
   When multiple people share an account, it is often difficult or impossible to know “who did what” using that account. All actions must be traced back to an individual, even when a shared account is used. This can be accomplished by tracking when individuals gain access to a shared account, as well as recording all actions during that session.

3. **Monitor the actions of privileged accounts to detect anomalous activities.**
   Unexpected activities can be an indication that an account has been breached. Organizations should look for spikes in activity from an account, large-scale accesses of data, logins from new locations, off-hours activity, etc.

4. **Establish an automated alerts and reporting mechanism.**
   Analytics need to be tied to actionable alerts and reports that document what happened, when it happened and by whom. Suspicious activity should set off alerts that are sent to the appropriate individuals.

5. **Tag all activities to facilitate audit record reviews.**
   All actions using a privileged account should be tied to the business owner of the asset. This facilitates business reviews by the appropriate people while eliminating unnecessary information that can hide unauthorized activities.

Step 3: Control what users and applications can do with privileged accounts.

1. **Control individual actions when using shared accounts.**
   More than one person often needs to access a system account that is often shared, however these individuals require specific access rights that are often unique. All actions using a shared account should be granted or denied based on the needs of the individual accessing that account. These controls should be done at the OS kernel level to ensure that they cannot be turned off or otherwise bypassed by a privileged administrator.

2. **Ensure local administrative sessions are controlled.**
   Some privileged identity management systems control administrative sessions through a proxy. These solutions can be bypassed by anyone with physical access to a target system. High-risk environments require that privileged accounts be protected from all logins, whether they are remote or at a local terminal. Security controls should not vary by login method.

3. **Protect key resources from even privileged accounts.**
   The most sensitive data should be protected from access by even administrators with authorized use of privileged identities on the system. This includes not only sensitive customer or internal data but system resources, such as the Windows Registry, directory controls, and binary integrity controls. This is a countermeasure to the "Lack of Self-Protection" security flaw.

4. **Apply application jailing.**
   Define accepted actions for high-risk applications to prevent an attacker from executing unauthorized commands. Any behavior that exceeds these bounds will be restricted by an application jailing function. For example, an ACL can be built based on a logical ID which owns Oracle processes and services so that its jailed behavior prohibits it from any actions other than starting Oracle Database Management System (DBMS) services.

Step 4: Apply complementary controls.

1. **Apply privileged identity controls to virtual environments.**
   Virtualization adds a new type of administrator—the hypervisor administrator. This class of administrator has the ability to copy or delete all virtual machines running on that hypervisor. Organizations should apply the same standard of controls over this class of administrator, including password controls and least-privilege access rights.

2. **Centrally manage all privileged identities using a single user store.**
   UNIX and Linux systems rely on user accounts on each local system to provide access and access controls. This leads to management oversights and errors. UNIX and Linux users should be managed from a central location, such as Microsoft Active Directory® to facilitate consistent, error-free administration.

3. **Implement advanced privileged identity governance.**
   Implement a program for governing privileged identities that goes beyond identifying orphaned accounts and privilege creep. In high-risk environments, an organization should understand what people can access, how they can get access, how they can control that access and how people are actually using their access. Required capabilities include: automated entitlement certification for users, roles and resources, centralized segregation of duties policies, monitoring of access rights with reports and dashboards and analysis of access rights and proposals for roles based on access patterns and organizational characteristics.

Making This Work

The problem with dealing with privileges is that it's hard. In even a small organization, many people need many different types of access to numerous systems and data. That is why a maturity model is needed. Organizations need to understand the security risks for their environments and apply the set of privileged identity management capabilities that best fit their requirements. With the right approach, improving the security of privileged accounts is a completely manageable task.

When starting a privileged identity management program, it can be easy to feel overwhelmed. People often wonder how they can make the program successful, from introducing their users to new restrictions, to changing the way that people work in order to fit a new security model.

While any program depends on the specific organization, from capabilities to culture, CA often recommends implementing a phased approach, both for the type of systems being secured and for the capabilities implemented. For example, many of our client organizations secure their environment in the following order:

1. UNIX/Linux and Windows systems
2. Databases
3. Network Devices
4. "Everything Else," including virtual environments

A typical implementation is often performed in the following order according to capabilities:

1. UNIX authentication bridging allows UNIX and Linux users to authenticate using their active directory credentials. Using a UNIX authentication bridge can eliminate the need to manage identities in UNIX/Linux environments by consolidating the distributed identities into an active directory. This is a good first step to any implementation because it centralizes the management of identities which makes the next steps more manageable.
2. Shared account password management controls access to privileged accounts. It stores passwords in a central location and helps provide accountability for user actions through secure auditing. This is often the second step in a privileged identity management implementation because it is a highly-visible "quick win" that provides significant security benefits while simplifying account access for administrators.
3. Reporting and auditing provides information that can be used to detect a security breach and can save significant time during a compliance audit. After access to privileged accounts is controlled, an organization should implement their required reporting and auditing capabilities.
4. Fine-Grained access controls/host security goes beyond OS-security to examine a user's original identity to determine whether an action should be allowed or denied. This enables true least privilege access. This step adds highly-effective security to an organization's most critical systems but should be implemented once organizational buy-in has been achieved with a series of quick wins.

CA Privileged Identity Manager

CA Privileged Identity Manager provides a comprehensive solution for privileged identity management in both physical and virtual environments. It provides a proactive approach to securing sensitive information and critical systems without impacting normal business and IT activities. CA Privileged Identity Manager helps to mitigate risk and facilitate compliance by controlling how privileged users access and use enterprise systems and data across the IT environment in order to achieve a higher level of security, reduced administrative costs and allow for easier audit/compliance processes.

CA Privileged Identity Manager is the only solution to implement access controls at the OS kernel level that is significantly harder to bypass than competing "sudo" and proxy-based solutions. It has a highly-scalable architecture that has been tested to run on over 100,000 endpoints.

Connect with CA Technologies at ca.com

CA Technologies (NASDAQ: CA) creates software that fuels transformation for companies and enables them to seize the opportunities of the application economy. Software is at the heart of every business, in every industry. From planning to development to management and security, CA is working with companies worldwide to change the way we live, transact and communicate – across mobile, private and public cloud, distributed and mainframe environments. Learn more at **ca.com**.

 CS200-135027_0615