

# Secure Management

of Privileged  
and Service Account  
Passwords



Every IT asset has at least one local, privileged login account. This includes workstations, servers, network devices, databases, applications and more. Some assets also have privileged accounts used to run services or authenticate one application to another.

Passwords for privileged accounts are used to install software, manage the device and perform technical support functions. They are often “all powerful,” having unlimited access to system functions and data. Consequently, compromise of privileged passwords is effectively compromise of the device.

Secure management of privileged passwords is essential to IT security. This document identifies technical challenges and offers solutions for effectively managing large numbers of sensitive passwords.

# Contents

- 1 Overview: The Business Problem** **1**
  
- 2 A Simple Solution: Randomize Passwords** **2**
  
- 3 Technical Challenges / Solution Requirements** **3**
  - 3.1 Platform Support . . . . . 3
  - 3.2 Workstations: Location and Connectivity . . . . . 3
  - 3.3 Scalability to Millions of Credentials . . . . . 4
  - 3.4 Reliable Operation and Race Conditions . . . . . 4
  - 3.5 Fault Tolerance: Hardware, Network and Facility Problems . . . . . 4
  - 3.6 Encryption in Transit and Storage . . . . . 5
  - 3.7 Connectivity and Firewalls . . . . . 5
  - 3.8 Services and Applications . . . . . 5
  - 3.9 Access Controls . . . . . 6
  - 3.10 Audit Trails and Alerts . . . . . 6
  
- 4 Architectural Elements** **7**
  - 4.1 Platform Support . . . . . 7
  - 4.2 Workstations: Location and Connectivity . . . . . 7
  - 4.3 Scalability to Millions of Credentials . . . . . 7
  - 4.4 Auto-discovery and Auto-configuration of Resources . . . . . 7
  - 4.5 Reliable Operation and Race Conditions . . . . . 8
  - 4.6 Fault Tolerance: Hardware, Network and Data Center Problems . . . . . 8
  - 4.7 Encryption in Transit and Storage . . . . . 8
  - 4.8 Connectivity and Firewalls . . . . . 8

4.9	Services and Applications . . . . .	9
4.9.1	Managing Passwords for Service Accounts . . . . .	9
4.9.2	Managing Application Passwords . . . . .	9
4.10	Access Controls . . . . .	10
4.11	Audit Trails and Alerts . . . . .	10

## 1 Overview: The Business Problem

In a typical enterprise-scale organization there are thousands of servers, workstations and network devices. Normally, there is a single, shared administrator password for every type of device. For example, one password may be used for each workstation of a given type or for every server with a given configuration. This is convenient for data center and desktop support staff: if they need to perform maintenance or an upgrade on a workstation or server, they know how to log in.

Such static and well-known privileged passwords create both operational challenges and security problems:

- When administrator login IDs are shared by multiple IT users, there is no audit log mapping administrative changes to individual IT staff. If an administrator makes a change to a system that causes a malfunction, it can be difficult to determine who caused the problem.
- When the same privileged ID and password exists on many systems, it is hard to coordinate password changes. As a result, privileged passwords are rarely changed and are often known to ex-employees.

These problems create security vulnerabilities. For example, if administrator passwords don't change, then former IT workers retain them beyond their term of employment. This clearly violates internal controls: former employees should not have administrative access to corporate systems.

In most organizations, strong internal controls are mandatory. Privacy protection legislation such as HIPAA and GLB, as well as legislation regarding corporate governance such as SOX, requires that systems containing sensitive data be secured against unauthorized access. Effective management of privileged passwords is therefore not an option, but a requirement.

## 2 A Simple Solution: Randomize Passwords

The obvious way to eliminate static and shared privileged passwords is to change them regularly. If every sensitive password were randomized daily, control problems would be alleviated.

Since IT users often need to use privileged passwords, randomizing passwords is only half of the solution. Additional functions are required to control access by IT users to changing passwords:

1. Authentication of IT users who wish to gain privileged access to a system.
2. Access control over which accounts IT users may access and when.
3. Audit logs recording such access, to create accountability.

The combined solution, capable of both randomizing large numbers of passwords and controlling access to password values or to the underlying accounts, can be complex. The following section describes some of the technical challenges that must be overcome in order to successfully deploy such a solution.

## 3 Technical Challenges / Solution Requirements

Describing a basic process for periodically randomizing and archiving administrator credentials is easy, while implementing such a process in a manner that scales well to thousands of devices, that is secure and fail-safe can be challenging.

The following sections describe some of the technical challenges such a system must address.

### 3.1 Platform Support

Every type of IT asset has a local administrator password. This is true even if network credentials are used in the normal course of business to manage the device, since a local administrator password must be used to attach each device to the network in the first place.

To be effective, a system for managing administrator passwords should support a broad array of platforms. This includes workstations, Windows servers, Unix servers, network routers, database servers, ERP applications, midrange servers (iSeries, VMS, etc.), mainframe computers, directories and more. In short, every device that contains sensitive data or whose operation is critical to the business should be supported.

### 3.2 Workstations: Location and Connectivity

A password management system can easily make connections to servers, which have fixed network addresses, are always on and are continuously connected to the network. It is much harder for a central password management server to connect to workstations, for several reasons:

- Workstations are mobile. Notebooks in particular frequently move from site to site.
- Even when they remain in one place, workstation IP addresses may change dynamically, due to use of DHCP.
- Workstations are often turned off and do not respond to network inquiries when deactivated.
- Workstations may be unplugged from the network, either to move them or for periods of disuse.
- Workstations may be protected by a firewall that blocks network connections inbound to the workstation.

In short, while it is easy for workstations to contact a central server, it is nearly impossible for the reverse to happen.

To reliably secure local administrator passwords on workstations, a password management system should include technology to overcome location, connectivity, address and firewall challenges.

### 3.3 Scalability to Millions of Credentials

A large organization may have thousands of workstations, servers and applications. If each of these IT assets gets a new administrator password daily, the total number of passwords that must be securely managed, including historical data, quickly grows into the millions of passwords.

Note that historical passwords need to be stored along with current ones, since in the event that a managed device crashes and is restored from backup media, its old password will be needed.

A scalable solution for managing administrator passwords must be able to randomize tens of thousands of passwords daily and to keep permanent records of millions of historical passwords.

### 3.4 Reliable Operation and Race Conditions

A robust system for managing administrator passwords must ensure that the password kept in its database for a given administrator account always matches the password on the system in question. This should be true even if an attempt to change passwords failed in the middle of an update.

For instance, if a password management system sets a new password on an IT asset and experiences a connection failure, it is not clear whether the new or old password is actually in effect – should the value stored in the database be updated?

A robust system for managing administrator passwords must ensure that the password it stores in its database is always the right one – even if a fault occurred in the middle of a password update.

### 3.5 Fault Tolerance: Hardware, Network and Facility Problems

A password management system must be fault tolerant. If it becomes unavailable, IT workers would not be able to do their jobs – making failure of the system catastrophic.

Hardware servers, including “appliances”<sup>1</sup> sometimes fail, due to disk crashes, power supplies burning up, etc. Network connections, especially over wide area links, also sometimes fail. Whole data centers can fail as well, due to power outages, earthquakes, hurricanes, tornados, fires or floods.

If one component of a privileged password management system fails, the passwords it manages must still be available. This is typically accomplished by running at least two servers, ideally at different sites. This means that if one server or one data center goes offline, IT staff elsewhere will be able to keep retrieving passwords and doing their jobs.

Fault tolerance between servers and sites requires data replication between servers. Such data replication must take place in real-time. The alternative – scheduled, batch replication – is inadequate. Consider, for example, a backup system that runs nightly. If a password management server were to fail just before a backup cycle begins, then the day’s new passwords would be lost. If passwords are changed daily, the current administrator password for almost every system would be lost: a catastrophic event.

---

<sup>1</sup>Appliances are generally just branded x86 servers.

### 3.6 Encryption in Transit and Storage

Compromise of a privileged password represents business risk. Compromise of many privileged passwords may represent catastrophic business risk. Consequently, a system for managing privileged passwords must protect these passwords cryptographically. It should protect passwords both when they are stored (at rest) and in transit: between users and itself, between replicated servers and between itself and target devices.

### 3.7 Connectivity and Firewalls

Networks are increasingly being segmented in order to create a layered defense against intruders. This creates situations where the password management system is attached to one network segment while an IT asset with privileged passwords is attached to another segment.

To manage passwords on a system on the far side of a firewall, a password management system must be able to send password updates over the firewall. This may not be simple: many network protocols are insecure by design (e.g., SMB for Windows, SQL\*Net for Oracle, plaintext LDAP, plaintext HTTP, etc.) and are blocked by firewall administrators for good reason.

To overcome this problem, an effective password management system must be able to replace network protocols that are native to a given target system with its own protocol. The password management system's network protocol must be appropriate to pass over a firewall.

### 3.8 Services and Applications

Sensitive passwords are not limited to those used by human IT workers. There are also service accounts, used to run attended software such as web servers and application passwords. There are also application passwords, used by one service on one computer to authenticate itself to another service, possibly on another computer.

On many systems, service passwords are static and application passwords are embedded in scripts, programs or text files. These passwords unlock login IDs that are often just as powerful as administrator accounts.

An effective solution for managing sensitive passwords should include mechanisms for managing service and application passwords, in addition to managing the administrator passwords used by IT workers. This calls for two specific capabilities:

1. The ability to automatically notify one program of the new password it should use to run a second program, after the password on the account used to run the second program has been randomized.
2. An API that allows one application to securely fetch a password that it can subsequently use to authenticate itself to another application.

### **3.9 Access Controls**

Not every IT worker should be able to access every privileged account. Likewise, applications invoking an API to retrieve a password should only be able to get passwords for services to which they legitimately need to be able to connect.

To enforce such security policies, a password management system must include a flexible access control infrastructure, capable of determining whether a given user of the system – human or software agent – should be granted access to a given privileged account.

### **3.10 Audit Trails and Alerts**

Every action in the password management system, including looking up assets and their passwords and changing access control policies should be auditable. This creates a chain of accountability between users and their actions.

It also makes sense to link auditable events to alerts. For example, if a legitimate user retrieves a given server's administrator password, the owner of that server might wish to receive an e-mail about the event.

To create accountability, to meet audit requirements and to enable system owners to promptly respond to anomalous administrator activity, a privileged password management system must include detailed logs of access to passwords, must retain its audit data indefinitely and must be able to act on, rather than just record, security events.

## 4 Architectural Elements

Each of the requirements set forth in the previous section can be addressed with a suitable architectural element in the password management solution. These architectural components are described in the following sections:

### 4.1 Platform Support

A rich set of connectors should be provided, to integrate with a broad range of target system types.

### 4.2 Workstations: Location and Connectivity

Client software should be available, to be installed on user workstations, which periodically contacts a central cluster of password management servers and requests new passwords for locally managed accounts.

This “pull mode” approach eliminates the problems with a central server “pushing” out passwords to devices with intermittent connectivity and dynamic IP addresses.

### 4.3 Scalability to Millions of Credentials

Multiple, concurrently-active password management servers should be supported, each of which can push new passwords to servers and each of which can provide new passwords to workstations on demand.

As the need for scalability grows, the number of servers can be increased. Servers should be placed behind a load balancer to hide this complexity from users and workstations.

### 4.4 Auto-discovery and Auto-configuration of Resources

It is not feasible to manually configure thousands of devices for periodic password changes. Instead, a privileged password management system requires an auto-discovery infrastructure to:

1. Automatically find servers and workstations.
2. Automatically find administrator and service accounts.
3. Configure systems and accounts for periodic password updates.
4. Notify software components of new service account passwords.

## 4.5 Reliable Operation and Race Conditions

A reliable protocol is required, especially for workstations, to confirm password updates before updating stored passwords.

Historical passwords should be retained indefinitely. In the event that an IT asset was damaged and had to be recovered from backup media, passwords from the date the backup was made will be available.

## 4.6 Fault Tolerance: Hardware, Network and Data Center Problems

As mentioned in [Subsection 4.3](#) on [Page 7](#), multiple servers are required. Not only should the servers each be able to randomize passwords in a multi-master configuration, but each server should house a complete data set and should replicate all local updates to that data to every other server.

Multiple servers should be installed in different data centers. This provides the opportunity for performance tuning, by having a local server manage passwords on local assets. It also provides for fault tolerance in the event of a disaster at one data center. If one data center goes offline, the password management servers at other data centers can keep working and will contain a full data set.

## 4.7 Encryption in Transit and Storage

Design of an encryption system for a password management system revolves around key management: How are keys generated? How are keys associated with data, with servers, with end users and with managed devices? Key management is an advanced topic and deserves separate treatment, beyond what this white paper can cover. That said, some basic observations can be made:

1. Users can sign into the system with a user interface carried over HTTPS – i.e., HTTP over SSL.
2. Connections between the password management system and target servers will generally use their native protocols, whose security will range from strong (e.g., HTTPS, SSH or LDAPS) to weak (e.g., SQL\*Net, LDAP). External measures, such as IPSec, may be appropriate to protect communication with some targets.
3. Connections between workstations and the password management system may be encrypted using HTTPS or using another key handshake protocol.
4. Connections between multiple password management servers may be encrypted using either SSL – which requires one cryptographic certificate to be purchased per server – or using symmetric server keys generated for each server.

## 4.8 Connectivity and Firewalls

In order to cross firewalls without exposing insecure protocols, the password management system must have components on both sides of the firewall. To avoid the need to fragment password storage into one

database per network segment, it makes sense to provide a proxy server – i.e., a server installed on one network segment whose purpose is to run connectors and update passwords on another network segment.

The communication between a primary password management server and a password management proxy server can be a simple, encrypted protocol over an arbitrarily numbered TCP port. This is robust, secure, bandwidth efficient and easy for firewall administrators to understand and forward.

## 4.9 Services and Applications

### 4.9.1 Managing Passwords for Service Accounts

In order to manage passwords used to start services, the password management system must be able to execute plug-in code, after successfully randomizing a password. The function of this installation-specific code is to notify network components of the new password value.

Some plug-ins are common. For example, the Windows Service Control Manager, Scheduler and IIS web server all store passwords in secondary storage (outside of the security database) in order to execute processes as named users. Since other programs may have the same requirement, the infrastructure for notifying programs of new passwords must be extensible (hence plug-ins).

### 4.9.2 Managing Application Passwords

In order to manage passwords used by one application to authenticate to another, an API must be exposed, to enable applications to acquire current credentials. For example, a web application might use the API to get a database password and use that password to connect to a database and read data which is then displayed in a web page.

This type of API creates a circular problem: how does an application which needs a password authenticate itself to the password management system? The obvious answer is that it must have its own (static) password, but this approach is clearly undesirable, as it reduces security of the application password (now randomized) back to a static password – but the point of a privileged password management system is precisely eliminate static password.

Some options for authenticating applications to the API include:

1. Using one-time passwords. The API can return not only the desired password, but also a new password which the calling application must use on for its next authentication.
2. Using environmental characteristics of the calling application. For example, a given application may only be allowed to sign into the API if it connects from a given IP address, or from a device running a particular operating system version, or even from an executable with a specific checksum.

## 4.10 Access Controls

A simple access control model maps privileges between individual passwords and individual users. For example, user X is allowed to retrieve the current password for login ID Y on system Z.

As the number of systems, managed user accounts and IT users grows, this model breaks down – there are simply too many relationships.

A more powerful model is to insert security groups between users and resources. Essentially users are collected into groups (each user can belong to multiple groups) and groups are assigned privileges to groups. For example, users A, B and C belong to group G. Members of group G are allowed to retrieve the current password for login ID X on system Y and login ID Z on system W.

This model may also be difficult to manage in large environments – users must be explicitly attached to groups (an administrative burden where there are many users and their responsibilities change often) and large numbers of resources must be manually attached to multiple groups.

The best model is to define both user groups and resource groups and to define access controls (privileges) between the two types of groups. For example, users A, B and C belong to user group UG1. Resources R, S and T belong to resource group RG1. Members of user group UG1 are allowed to retrieve passwords for accounts in resource group RG1.

This model provides for maximum flexibility and minimum administrative burden. It can be optimized further by automating association of users with user groups and resources with resource groups.

1. User membership in resource groups can be determined based on their identity attributes or group memberships in a corporate directory (LDAP or Active Directory).
2. Resource membership in resource groups can be determined based on characteristics of the resource – for example based on IP address, hardware class, operating system, MAC address, directory OU of the system's representative computer object, etc.

## 4.11 Audit Trails and Alerts

Logging is straightforward – record every event as it takes place and provide reports that are either user-centric or resource-centric to show event history.