Delinea

How to Apply Zero Standing Privilege Consistently Across Windows and *NIX Systems

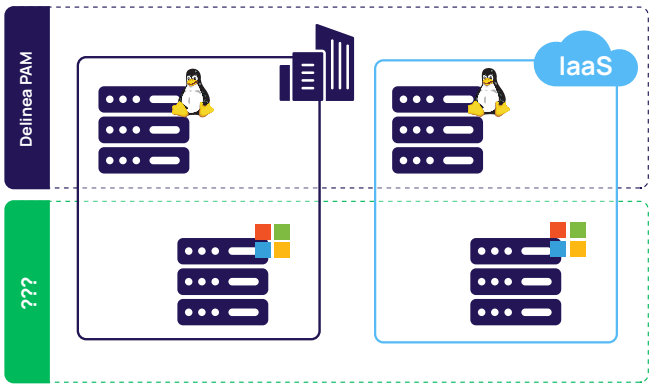
Cloud transformation projects abound, with enterprises in all industries deploying new application workloads in single- and multi-cloud environments. As a result, the spotlight focuses on the increasing data security challenges introduced by a distributed IT infrastructure with new attack surfaces.

Couple this with a growing number of administrators (internal and third-party), systems (physical and virtual), and credential-based breaches — a privileged access management (PAM) solution with a "least privilege" approach at its core is a critical best practice.

While most organizations understand and subscribe to least privilege guidance on a rational level, implementation is, unfortunately, fragmented. The biggest obstacle is that IT is often functionally siloed, with independent Windows and Linux/Unix (*NIX) teams, each with autonomy over decisions involving their respective systems' security. This results in PAM controls that are inconsistent and where a weaker side exposes the broader IT infrastructure to greater risk.

Delinea has written this solution brief for decision-makers responsible for ensuring the security of Windows systems and the data they contain. Currently, you rely on native Windows and Active Directory security controls and tools to govern access, while your peers who own the *NIX estate use the Delinea Server Suite.

Microsoft and Delinea designed their respective solutions to prevent breaches that exploit privileged accounts as the primary attack vector. But, as this solution brief will demonstrate, the Windows side of the equation introduces unacceptable risks.



The challenge we face

Most companies we talk to get it. IT understands that least privilege is the best practice approach to privileged access management.

However, the big challenge is not recognizing the need but instead deploying it properly and effectively across the entire IT server infrastructure. Over the last 16 years, Delinea has experienced this a lot. However, it's often the

*NIX team that reaches out to Delinea. Even though its native "sudo" implements a privilege elevation model, *NIX systems lack centralized role-based management and fine-grained authorization, resulting in complexity, management overhead, security gaps, compliance issues. So, for the *NIX team, we solve this by bringing their servers under Active Directory's management model, extending it to be more effective and scalable, and introducing more powerful access controls on the *NIX servers.

In contrast, the Windows team generally relies on native Windows and Active Directory security and tools. We find that IT assigns admins full access rights as a standard practice. In this context, we've identified three common approaches:

- **Default Local Administrator Account Login:** IT gives admins the password to the local Administrator account to either log in directly or "Run as administrator" and satisfy any User Access Control (UAC) prompts for elevated access whenever they want unconstrained.
- **"Run as" The Dash-A Account:** Like "Run as administrator," this allows the admin user to specify an alternate account to use, in this case, their highly privileged dash-a account (see later for more on dash-a).
- **Local Administrator Group Membership:** As a member of this privileged Active Directory security group, upon login, the user inherits the privileges of that group.

Others may use Group Policy and attempt to apply security controls more selectively to users and computers along with "Run as…" However, none of these approaches abide by a least privilege approach to security.

As we'll discuss, this, along with other native tools, results in a ton of complexity, inconsistent security, privilege creep, and a reluctance to mess with it for fear of disruption. Business never stops, however. As more Windows instances spring to life in the cloud, the hole only gets deeper.

Why native Windows security and Active Directory tools are not enough

Some of the most common native approaches involve Group Policy (GP), admin security group membership, alternate admin ("dash-a") accounts, Local Administrator Password Solution (LAPS), Microsoft App Locker, and the "Red Forest" security architecture. Let's look at each, starting with GP, which is the most common and frequently used approach.

Group policy

Ordinarily, IT would need to log in to each computer to configure its security and operational behavior locally. The Active Directory Group Policy (GP) tools allow IT to do all this centrally and have the settings automatically pushed to each machine. Through GP Objects (GPOs), IT can grant users job-appropriate rights, give them access to services and applications, and implement security policies such as strong password strength. The job-appropriate rights part is of concern, though. How to control what the user can access?

There are many moving parts to GP, making it very difficult to design, implement, troubleshoot, and audit. One of the most complicated aspects of GPOs is security filtering (essentially, Access Control Lists on the GPO). This can be very complex, easy to get wrong, and the UI itself makes it very hard to determine comprehensively what's set and how they interact and affect each other.

When making changes in hopes for a desired outcome, this complexity can easily cause a GP to break, leading to unexpected results — security holes, lack of system/resource availability, and productivity issues — and frustrations trying to debug the problem. It's easy to grant any user access to any file or application on any computer. However, setting the wrong permissions in a GPO can easily result in a user not being able to login in the morning or, perhaps worse — giving the user too much access.

Especially in larger environments, this becomes impossible to manage, as it requires a lot of manual configuration, and the complexity makes it easy to lose track of the scope. For example, adding a multitude of GPOs at different levels both locally and at a domain level within Active Directory can result in unexpected results due to a GPO, somewhere, with higher precedence.

The most common approach is to group users and computers into an Org Unit (OU) and apply GPOs to the OU. This can result in dozens of OUs, often nested and very messy. Adding a new

GP for a specific computer affects all in the OU, or else, requires yet another OU with that Active Directory GP applied, ultimately duplicating and fragmenting the security model.

Not surprisingly, many customers adopt a "set it and forget it" approach, configuring a generalized set of GPOs and not touching them again for fear of breaking things. This might be the most significant barrier to change and one that is hard to overcome without some compelling event such as a breach or a new CIO who understands the benefits of an identity-centric PAM solution designed to solve this and other related security issues.

Admin security group membership

Since using Group Policy to scope administrator rights accurately is challenging, customers may rely on simply adding the user to a local Administrators group, giving them full rights to log in and do anything on the box. Hire a new administrator — just add them to the group and let them get on with their work unhindered.

This is not least-privilege, and if a cyber attacker compromises the account, they inherit full rights on the box. Once the attacker gains a foothold with admin-level privileges, they can use many open- source tools such as Mimikatz. They can then extract authentication credentials (for example, Kerberos tickets) for other accounts that have used the machine, such as a domain administrator account, allowing them to spread laterally within the network.

Dash-A accounts

Giving admins an alternate admin account is another approach to minimize the potential harm that might come if an attacker compromises an administrator's regular account.

With this approach, a standard user account (e.g., jane.doe@ acme. com) would have minimum rights, enough for routine end-user tasks such as surfing the Web, running productivity apps like Office 365, and doing email. The dash-a account (e.g., jane.doe-admin@ acme.com) would be the highly privileged one used to log into infrastructure servers and perform privileged tasks.

The problem with this is that the dash-a is rarely constrained, giving administrators unnecessary rights beyond their job function scope. It's typically assigned static, domain-level rights, perpetuating the use of standing privileges.

Administrators on desktops will frequently run desktop tools such as Active Directory Users and Computers (ADUC) via the "Run as" command, selecting their dash-a account for full domain rights to make Active Directory changes. This has led to the proliferation of attacks such as pass-the-hash, which relies on an attacker stealing cached credentials from a compromised system — often, a Windows desktop.

Using dash-a accounts routinely increases the potential impact of a mistake. It also increases the attack surface, giving a threat actor a stronger position to move laterally in the network.

Local Administrator Password Solution (LAPS)

Local Administrator accounts have been, and continue to be, a significant source of risk for most networks. There's no native or straightforward way to manage them once IT has deployed a system in the live environment. Often, they share the same password across systems. Therefore, cyber adversaries target local accounts and use them to move laterally between systems.

When moving between systems whose local accounts share a password, the cyber adversary can bypass many defenses since no domain authentication is required. This leaves a massive hole in

your defensive strategy. Unfortunately, there's no easy, cost-effective way of managing these local accounts while maintaining a strong security posture.

Microsoft is well-known for filling a void with a free tool that has limited functionality. LAPS is one such tool widely deployed to its credit. However, when requirements inevitably extend beyond its basic capabilities, the customer is forced to augment with other tools, resulting in a mishmash.

LAPS introduces a new client-side extension installed on each machine that launches every time a GP refresh occurs. It rotates the local Administrator account password and stores it as an attribute in the Computer Account object in Active Directory (requiring a schema extension). On the positive side, this ensures a different password for the local Administrator account on each system (to combat lateral movement).

Limitations, however, include:

- Active Directory stores the password unencrypted.
- The clear text password only has an ACL to protect it, and a threat actor can easily access it if delegation is not adequately planned or deployed.
- Only one local administrator account can have its password managed by LAPS at a time. Any other custom privileged local accounts remain at risk.
- If a threat actor compromises the Domain Controller, s/he can compromise all local administrator account passwords in that domain.
- Auditing of privileged activity is weak, and IT must configure it per OU/per group. It requires (often complicated) event correlation to piece together an attack sequence. There's no centralized log of who updates a password and when, and no native session recording capability.
- Admins can still rotate the local machine passwords without the Active Directory being aware of the changes, leaving the host and LAPS entry out of sync. Similarly, there's no uniform password reconciliation mechanism to bring passwords back under control.
- GP does not automatically rotate the password after use but on a schedule, exposing a larger attack window.
- There's no native workflow-based access request/approval mechanism to govern access to the passwords, no way to time- box access with automatic revocation, nor any native multi-factor authentication (MFA) on access.
- There is no password history, so if you restore a machine to a point older than the current password requires, you need to obtain the correct password manually via other tools. Similarly, Microsoft recommends disabling startup repair on all machines as it may revert the machine to an earlier restore state with a different administrator password.
- This mechanism will not extend to non-Windows systems in your infrastructure, requiring different technology, policy management, and IT/operations processes.
- IT uses PowerShell commands for some LAPS administration tasks, but Microsoft has not installed it natively on every machine.

However, IT needs to use PowerShell tools to list all non-compliant accounts.

- LAPS is not recommended for Domain Controllers, as it will change the domain's Administrator account instead of the actual local Administrator account (the DSRM account).
- LAPS requires the machine to be on the domain, thus excluding any non-domain systems.

	Delinea	MS LAPS
Local admin accounts	✓	✓
Domain admin, service, app, & DB, accounts	✓	✗
Workflow-based access request	✓	✗
MFA for password retrieval	✓	✗
Centralized session monitoring, recording, & reporting	✓	✗
Automatic password reconciliation	✓	✗
Account discover	✓	✗
Password rotation after use	✓	✗
*NIX local accounts	✓	✗

Microsoft AppLocker

AppLocker is a tool designed to govern access to executables, DLLs, scripts, Windows installers, and packaged apps. The goal is to prevent unintentional system compromise by internal users or malicious software. From a pure privileged access management perspective, AppLocker has limited value aiming to stop administrators from accidentally running things they shouldn't. It will not stop admins who know what they're doing.

Essentially an application whitelisting tool, it's another basic Microsoft offering that satisfies minimal PAM use cases. The Active Directory GP framework manages and distributes

AppLocker policies. They specify the authorized apps that users may open on servers and workstations. It defaults to deny, so if a file is not on the whitelist, AppLocker won't open it.

The good news is that in theory, it's simple and effective; it's free with the enterprise edition of Windows Server; it's built-into GP, and it automatically whitelists internal Windows apps to save time. Just like LAPS, however, problems arise when the customer needs to satisfy non-basic use cases. It may be tempting to turn AppLocker into an application blacklist solution, but this requires a ton of work to block everything and maintain the extra policies. Again, a clever admin will always find a way to bypass this.

Some of the drawbacks include:

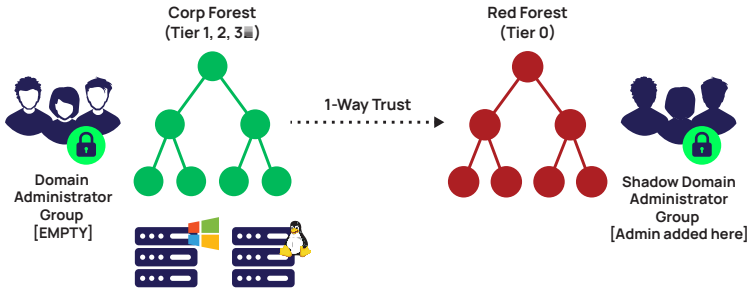
- The need to constantly tweak a company-wide set of policies as the list of users, systems, and applications changes (more so on workstations vs. infrastructure servers). This is an inherent weakness in this and any whitelisting solution.
- Anyone with local Administrator rights on a box can subvert AppLocker completely.
- AppLocker does not support security levels, so there's no fine- grained access control to scope rights differently for different users.
- No native workflow-based access control mechanism for users to request access.
- Access is static, i.e., no way to time-limit access with automatic revocation.
- There's no native MFA for additional identity assurance during file access.
- A typical policy is to block the execution of files in a specific folder. Administrators can move or copy the files to another folder where execution is permitted.
- Admins can use gpedit.msc on the local server and create their own rules to merge with the centrally controlled AppLocker.
- AppLocker doesn't play well with Windows 10 Professional and doesn't support *NIX. So, if your environment includes such systems, you'll need to look elsewhere for a solution.

Microsoft Enhanced Security Administrative Environment (ESAE)

Faced with a growing number of attacks such as "pass the hash" and "pass the ticket," after the release of Windows 2016, Microsoft introduced new domain architectures using native Active Directory and Microsoft Identity Manager (MIM) capabilities. The Enhanced Security Administrative Environment (ESAE), also known as "Red Forest," is the most well-known, designed to minimize the risk of a domain-level breach.

At a very high level, ESAE is a multi-tier architecture where IT places systems, users, and security groups into tiers. Tiers represent their relative value or sensitivity (e.g., Tier 3 for workstations, Tier 2 for databases, Tier 1 for enterprise servers, and Tier 0 for the Red Forest containing Active Directory domain-specific objects).

The Red Forest design provides many sound recommendations for improving security related to privilege. However, the nature of the architecture still leaves much to be desired. Administrators are still over-privileged; they must deal with multiple accounts, they're trusted to know when to use them and when not (i.e., trust is not time-bound), and the model does little to address systems not distributed by Microsoft.



One use of this model is to reduce the risks associated with the common practice of giving administrators standing membership — for example — the Domain Administrators group. It enables administration of the domain from the Red Forest instead of the corporate forest to reduce the risk of a full administrative compromise. A MIM self-service portal allows an admin to request membership of the group. Instead of provisioning the user into the corporate Active Directory forest, MIM provisions the user into a shadow Domain Administrator Group in the

Red Forest. Also, MIM grants access for a time-limited period. When the user then logs into a corporate domain-joined server, Windows will recognize the user's Red Forest shadow security group membership and give the user the privileges associated with the equivalent security group on the corporate side without explicitly adding the user as a member of it.

All this is reasonable in a pure Windows infrastructure. Still, as organizations grow and establish new workloads in the cloud, this model will not accommodate the introduction of non-Windows workloads. Almost without exception, third-party PAM solutions for *NIX servers don't help since they don't have the advanced Active Directory integration necessary to support such use cases.

So, when an administrator logs directly into a *NIX server with Active Directory credentials, the third-party PAM solution will query the corporate Active Directory for group membership, coming up empty since the membership doesn't reside there - it's on the Red Forest side. It's unable to do the complex stuff such as navigating the 1-way, cross-forest trust, or looking inside Kerberos tickets to obtain Red Forest group memberships.

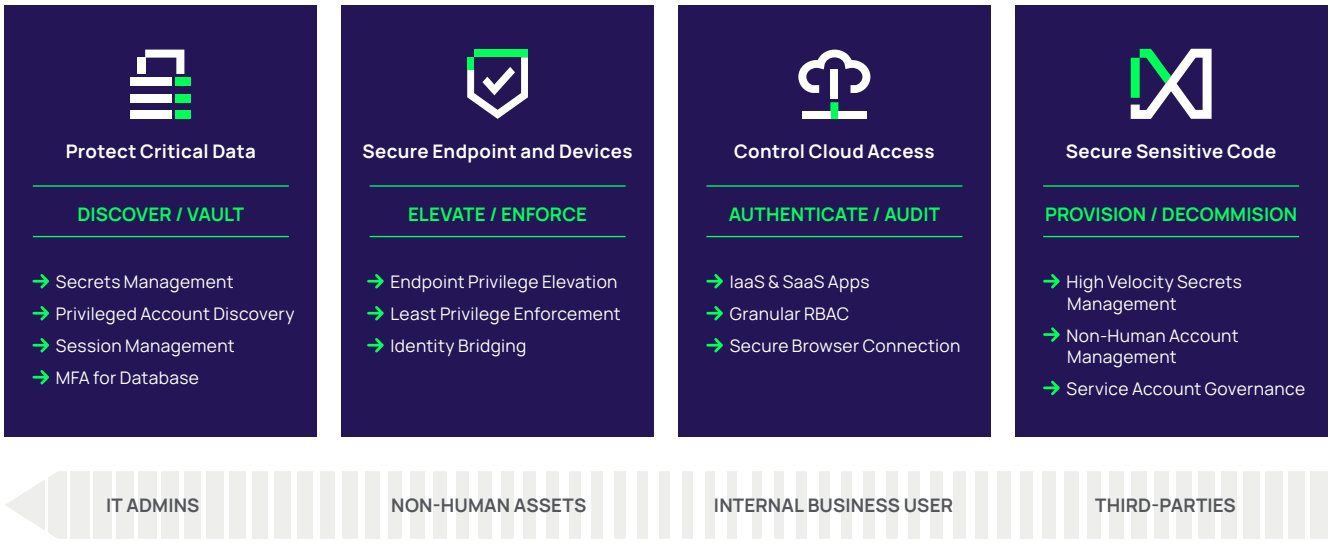
Similarly, when the admin logs into a *NIX server indirectly, from a Windows server, the PAM technology can't leverage the initial Windows login ticket to obtain the group memberships and pass them on to the user's *NIX session. The net-net is that *NIX systems are left out in the cold, resulting in additional risk of a breach, operational overhead, and compliance challenges.

Delinea PAM

Unfortunately for many organizations, they take the path of least resistance and patch together much of the above, resulting in dozens of native and open-source tools, management interfaces, and scripts. As the organizations grow and evolve, their use cases expand and become more sophisticated, which impacts their ability to contain risk to an acceptable level. Continuing down the same path inevitably leads to a disjointed and fragmented PAM infrastructure, resulting in higher risk, security holes, huge administrative overhead, additional cost, and compliance gaps.

Delinea's modern PAM resolves the complexities around infrastructure (Windows servers, Windows workstations, Linux, Unix, VMs), security (authentication and authorization), and cloud (virtual instances, containers, microservices). The power behind Delinea's approach is a common platform and a complete set of PAM capabilities on top of it. The platform's cloud-native foundations accommodate traditional data center infrastructures and modern hybrid cloud and multi-cloud infrastructures.

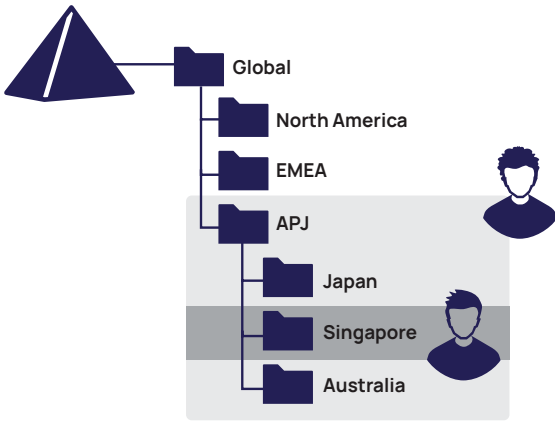
PAM Solutions Portfolio



As illustrated in the diagram, the platform makes available comprehensive PAM capabilities. Instead of coding things like MFA into each product module, they can each consume MFA-as-a-service consistently, wherever it's needed. This is in stark contrast to vendors whose offerings are a mix of home-grown, OEM'd, and acquired technologies that result in massive duplication, redundancy, and cost.

Delinea's solutions rely on implementing the best practice for PAM; least privilege with privilege elevation via the Delinea Server Suite. This allows customers to clinically scope administrative rights to the individual, aligning with their job function compared to full rights and carte blanch access to everything, which many native approaches perpetuate.

Delinea enables Active Directory to centrally manage users, roles, and rights that govern administrator access to Windows, Linux, and Unix. It layers a hierarchical "Zone" structure on top of the basic Active Directory container model with no schema changes, allowing customers to efficiently and effectively structure their governance model to align with the departmental structure, geography, or functional use such as a collection of HR servers, database servers, or PCI servers.



This enforces separation of duties and provides easy delegation of access control management to the teams responsible for the systems in each Zone. This radically simplifies access control management.

It ensures admins get access only to the systems they need to manage and only the specific privileges they need on those systems. For example, database admins can't touch web servers, and web admins can't touch databases.

So far, we've focused this discussion on human user-based interactions with IT infrastructure. This same model also extends to applications and services, ensuring that Delinea protects and centrally manages application and service accounts just like their human identity counterparts.

So, let's summarize the benefits of Delinea PAM compared to the native approaches highlighted above:

- Organizations leveraging Delinea avoid the complexities of relying on Group Policy for user rights management. Delinea roles managed centrally from Active Directory, define who has access to what and what they're allowed to do, as well as enforce MFA policy during login and privilege elevation. Just now, I said we avoid relying on Group Policy. The Group Policy framework does have its value, though. Hence, Delinea Server Suite can extend group policies to Linux and Unix systems, all centrally managed from Active Directory.

- Delinea locks down (ideally, disables) local Administrator accounts as a primary vector of attack. Administrators log in with their individual, low privileged account and elevate to run specific applications only when necessary. Workflow-based access request (from Delinea, as well as via integration with ServiceNow or SailPoint) for just-in-time access control avoids standing privileges and their associated risks. Approved roles are temporary and automatically revoked on expiration;

- Just as we now disable the use of local Administrator accounts, we similarly avoid using the local Administrators groups since membership grants the user (or the attacker) total ownership of the box. As mentioned above, admins log in as themselves with least privilege and elevate privilege only when needed.

- As an alternative to Microsoft LAPS, Delinea can shut down the primary vectors of attack — the local Administrator account and local Administrators groups, optionally vaulting default shared privileged accounts for break-glass emergency access only. Day- to-day, admins log in as themselves with a unique ID that ensures 100% accountability. Detailed auditing and visual recording of privileged session activity close the loop, providing incident

response and auditing teams with full visibility for root-cause investigation and compliance. This same Delinea model applies equally to non-Windows use cases for a centralized and consistent approach to privileged access security, enterprise-wide. Finally, suppose a rogue admin or attacker changes a local account password on the box. In that case, Delinea Vault Suite can detect the change and automatically reconcile it to maintain Privileged Access Service as the source of truth and avoid system outages.

- Delinea RBAC – centrally managed through Active Directory — along with host-level enforcement, bring true fine-grained access controls to the Windows server or workstation. This model consistently extends to *NIX and non-domain-joined servers, overcoming AppLocker's limitations and the inherent challenges of its application whitelist approach.

- Dash-a accounts can be retired. However, if they remain, we must avoid granting the alternate account full standing domain- level rights. Consistent with the above least privilege approach, it should have minimum rights with Delinea roles allowing the administrator to elevate privileges to run specific applications in line with that user's job function. As an extra protection, Delinea Vault Suite can discover dash-a accounts, securely vault them, automatically rotate them, and establish a login session without the password being revealed to the user; and

- An ESAE/Red Forest model An ESAE/Red Forest model can be daunting. But, for customers with an existing installation, Delinea's hierarchical Zone model provides isolation and segregation and can easily map to the customer's multi-tier ESAE model. It's more cost-effective, less complex, and extends to additional use cases (e.g., extending this model to *NIX and providing true least privilege access controls across the entire infrastructure). For *NIX use cases, Delinea honors shadow Red Forest security group memberships via Kerberos ticket interrogation and an ability to traverse complex Active Directory models such as the 1-way cross- forest trusts used in Red Forest deployments. As an alternative to Red Forest, however, Delinea offers a more robust and more straightforward architecture.

- In the original example earlier, Delinea mitigates the risks of adding users to a highly privileged security group such as Domain Administrators by eliminating such membership. This, plus leveraging a dash-a model, provides a more effective and simpler alternative to a Red Forest architecture. Consistent with a least privilege best practice, Delinea grants the administrator only the necessary rights to perform specific tasks (compared to full domain access) and can wrap access request/approval and MFA around this for additional governance and identity assurance.

Conclusion

There are many different native Microsoft tools organizations can use to solve discrete security issues in their Windows estate.

Unfortunately, this can easily result in a patchwork with inconsistent security, management, and reporting. This compounds when migrating business logic to the cloud and, further, when dealing with multi-VPC and multi-cloud workloads. Add *NIX into the mix, and organizations must now look outside the native Windows toolset to figure out a comparable set of security controls.

Delinea PAM is agnostic to an organization's platform and hosting infrastructure. It secures access to modern hybrid cloud infrastructures that include Windows, Linux, and UNIX servers running on traditional bare-metal hardware, virtual systems, containers, and microservices — be it on-premises or in single-, or multi-cloud architectures.

Delinea

Delinea is a leading provider of privileged access management (PAM) solutions that make security seamless for the modern, hybrid enterprise. Our solutions empower organizations to secure critical data, devices, code, and cloud infrastructure to help reduce risk, ensure compliance, and simplify security. Delinea removes complexity and defines the boundaries of access for thousands of customers worldwide. Our customers range from small businesses to the world's largest financial institutions, intelligence agencies, and critical infrastructure companies. **delinea.com**