Protecting Tier 0 the Modern Way

techcommunity.microsoft.com/blog/coreinfrastructureandsecurityblog/protecting-tier-0-the-modern-way/4052851

Blog Post

How should your Tier 0 Protection look like?

Almost every attack on Active Directory you hear about today – no matter if ransomware is involved or not – (ab)uses credential theft techniques as the key factor for successful compromise. Microsoft's State of Cybercrime report confirms this statement: "The top finding among ransomware incident response engagements was insufficient privilege access and lateral movement controls."

Despite the fantastic capabilities of modern detection and protection tools (like the Microsoft Defender family of products), we should not forget that prevention is always better than cure (which means that accounts should be protected against credential theft proactively). Microsoft's approach to achieving this goal is the Enterprise Access Model. It adds the aspect of hybrid and multi-cloud identities to the Active Directory Administrative Tier Model. Although first published almost 10 years ago, the AD Administrative Tier Model is still not obsolete. Not having it in place and enforced is extremely risky with today's threat level in mind.

Most attackers follow playbooks and whatever their final goal may be, Active Directory Domain domination (Tier 0 compromise) is a stopover in almost every attack. Hence, **securing Tier 0 is the first critical step towards your Active Directory hardening journey and this article was written to help with it.**

AD Administrative Tier Model Refresher

The AD Administrative Tier Model prevents escalation of privilege by restricting what Administrators can control and where they can log on. In the context of protecting Tier 0, the latter ensures that Tier 0 credentials cannot be exposed to a system belonging to another Tier (Tier 1 or Tier 2).

Tier 0 includes accounts (Admins-, service- and computer-accounts, groups) that have direct or indirect administrative control over all AD-related identities and identity management systems. While direct administrative control is easy to identify (e.g. members of Domain Admins group), indirect control can be hard to spot: e.g. think of a virtualized Domain Controller and what the admin of the virtualization host can do to it, like dumping the memory or copying the Domain Controller's hard disk with all the password hashes. Consequently, virtualization environments hosting Tier 0 computers are Tier 0 systems as well. This also applies to the virtualization Admin accounts.

The three Commandments of AD Administrative Tier Model

Rule #1: **Credentials from a higher-privileged tier** (e.g. Tier 0 Admin or Service account**) must not be exposed to lower-tier systems** (e.g. Tier 1 or Tier 2 systems).

Rule #2: **Lower-tier credentials can use services provided by higher-tiers, but not the other way around.** E.g. Tier 1 and even Tier 2 system still must be able to apply Group Policies.

Rule #3: **Any system or user account that can manage a higher tier is also a member of that tier, whether originally intended or not.**

Implementing the AD Administrative Tier Model

Most guides describe how to achieve these goals by implementing a complex cascade of Group Policies (The local computer configuration must be changed to avoid that higher Tier level administrators can expose their credentials to a down-level computer). This comes with the downside that Group Policies can be bypassed by local administrators and that the Tier Level restriction works only on Active Directory joined Windows computers. The bad news is that there is still no click-once deployment for Tiered Administration, but there is a more robust way to get things done by implementing Authentication policies. Authentication Policies provide a way to contain high-privilege credentials to systems that are only pertinent to selected users, computers, or services. With these capabilities, you can limit Tier 0 account usage to Tier 0 hosts. That's exactly what we need to achieve to protect Tier 0 identities from credential theft-based attacks.

To be very clear on this: **With Kerberos Authentication Policies you can define a claim which defines where the user is allowed to request a Kerberos Granting Ticket from.**

Optional: Deep Dive in Authentication Policies

Authentication Policies are based on a Kerberos extension called FAST (Flexible Authentication Secure Tunneling) or Kerberos Armoring. FAST provides a protected channel between the Kerberos client and the KDC for the whole pre-authentication conversation by encrypting the pre-authentication messages with a so-called armor key and by ensuring the integrity of the messages.

Kerberos Armoring is disabled by default and must be enabled using Group Policies. Once enabled, it provides the following functionality:

- Protection against offline dictionary attacks. Kerberos armoring protects the user's pre-authentication data (which is vulnerable to offline dictionary attacks when it is generated from a password).
- Authenticated Kerberos errors. Kerberos armoring protects user Kerberos authentications from KDC Kerberos error spoofing, which can downgrade to NTLM or weaker cryptography.
- Disables any authentication protocol except Kerberos for the configured user.
- Compounded authentication in Dynamic Access Control (DAC). This allows authorization based on the combination of both user claims and device claims.

The last bullet point provides the basis for the feature we plan to use for protecting Tier 0: Authentication Policies.

Restricting user logon from specific hosts requires the Domain Controller (specifically the Key Distribution Center (KDC)) to validate the host's identity. When using Kerberos authentication with Kerberos armoring, the KDC is provided with the TGT of the host from which the user is authenticating. That's what we call an armored TGT, the content of which is used to complete an access check to determine if the host is allowed.

Kerberos armoring logon flow (simplified):

1. The computer has already received an armored TGT during computer authentication to the domain.
2. The user logs on to the computer:
    1. An unarmored AS-REQ for a TGT is sent to the KDC.
    2. The KDC queries for the user account in Active Directory and determines if it is configured with an Authentication Policy that restricts initial authentication that requires armored requests.
    3. The KDC fails the request and asks for Pre-Authentication.
    4. Windows detects that the domain supports Kerberos armoring and sends an armored AS-REQ to retry the sign-in request.
    5. The KDC performs an access check by using the configured access control conditions and the client operating system's identity information in the TGT that was used to armor the request. If the access check fails, the domain controller rejects the request.
3. If the access check succeeds, the KDC replies with an armored reply (AS-REP) and the authentication process continues. The user now has an armored TGT.

Looks very much like a normal Kerberos logon? Not exactly: The main difference is the fact that the user's TGT includes the source computer's identity information. Requesting Service Tickets looks similar to what we described above, except that the user's armored TGT is used for protection and restriction.

Implementing a Tier 0 OU Structure and Authentication Policy

The following steps are required to limit Tier 0 account usage (Admins and Service accounts) to Tier 0 hosts:

1. Enable Kerberos Armoring (aka FAST) for DCs and all computers (or at least Tier 0 computers).
2. Before creating an OU structure similar to the one pictured below, you MUST ensure that Tier 0 accounts are the only ones having sensitive permissions on the root level of the domain. Keep in mind that all ACLs configured on the root-level of fabrikam.com will be inherited by the OU called "Admin" in our example.

3. Create the following security groups:

- Tier 0 Users

- Tier 0 Computer

4. Constantly update the Authentication policy to ensure that any new T0 Admin or T0 service account is covered.

5. Ensure that any newly created T0 computer account is added to the T0 Computers security group.

6. Configure an Authentication Policy with the following parameters and enforce the Kerberos Authentication policy:

(User) Accounts	Conditions (Computer accounts/groups)	User Sign On
T0 Admin accounts	(Member of each({ENTERPRISE DOMAIN CONTROLLERS}) Or Member of any({Tier 0 computers (FABRIKAM\Tier 0 computers)}))	Kerberos only

The screenshot below shows the relevant section of the Authentication Policy:

Find more details about how to create Authentication Policies at https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/how-to-configure-protected-accounts#create-a-user-account-audit-for-authentication-policy-with-adac.

Tier 0 Admin Logon Flow: Privileged Access Workstations (PAWs) are a MUST

As explained at the beginning of the article, attackers can sneak through an open (and MFA protected) RDP connection when the Admin’s client computer is compromised. To protect from this type of attack Microsoft has been recommending using PAWs since many years.

In case you ask yourself, what the advantage of restricting the source of a logon attempt through Kerberos Policies is: Most certainly you do not want your T0 Admins to RDP from their – potentially compromised – workplace computers to the DC. Instead, you want them to use a Tier 0 Administrative Jump Host or - even better - a Privileged Access Workstation. With a compromised workplace computer as a source for T0 access it would be easy for an attacker to either use a keylogger to steal the T0 Admin’s password, or to simply sneak through the RDP channel once it is open (using a simple password or MFA doesn’t make a big difference for this type of attack). Even if an attacker would be able to steal the credential of a Tier 0 user, the attacker could use those credentials from a computer which is defined in the claim. On any other computer, Active Directory will not approve a TGT, even if the user provides the correct credentials. This will give you the easy possibility to monitor the declined requests and react properly.

There are too many ways of implementing the Tier 0 Admin logon flow to describe all of them in a blog. The “classic” (some call it “old-fashioned”) approach is a domain-joined PAW which is used for T0 administrative access to Tier 0 systems.

The solution above is straightforward but does not provide any modern cloud-based security features.

“Protecting Tier 0 the modern way” not only refers to using Authentication Policies, but also leverages modern protection mechanisms provided by Azure Entra ID, like Multi-Factor-Authentication, Conditional Access or Identity Protection (to cover just the most important ones).

Our preferred way of protecting the Tier 0 logon flow is via an Intune-managed PAW and Azure Virtual Desktop because this approach is easy to implement and perfectly teams modern protection mechanisms with on-premises Active Directory:

Logon to the AVD is restricted to come from a compliant PAW device only, Authentication Policies do the rest.

Automation through PowerShell

Still sounds painful? While steps 1 – 3 (enable Kerberos FAST, create OU structure, create Tier 0 groups) of Implementing a Tier 0 OU Structure and Authentication Policy are one-time tasks, step 4 and 6 (keep group membership and Authentication policy up-to-date) have turned out to be challenging in complex, dynamic environments. That’s why Andreas Lucas (aka Kili69) has developed a PowerShell-based automation tool which …

- creates the OU structure described above (if not already exists)
- creates the security groups described above (if not already exist)
- creates the Authentication policy described above (if not already exists)
- applies the Tier 0 authentication policy to any Tier 0 user object
- removes any object from the T0 Computers group which is not located in the Tier 0 OU
- removes any user object from the default Active directory Tier 0 groups, if the Authentication policy is not applied (except Built-In Administrator, GMSA and service accounts)

Additional Comments and Recommendations

Prerequisites for implementing Kerberos Authentication Policies

Kerberos Authentication Policies were introduced in Windows Server 2012 R2, hence a Domain functional level of Windows Server 2012 R2 or higher is required for implementation.

Authentication Policy – special Settings

Require rolling NTLM secret for NTLM authentication

Configuration of this feature was moved to the properties of the domain in Active Directory Administrative Center. When enabled, for users with the “Smart card is required for interactive logon” checkbox set, a new random password will be generated according to the password policy. See https://learn.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/whats-new-in-credential-protection#rolling-public-key-only-users-ntlm-secrets for more details.

Allow NTLM network authentication when user is restricted to selected devices

We do NOT recommend enabling this feature because with NTLM authentication allowed the capabilities of restricting access through Authentication Policies are reduced. In addition to that, we recommend adding privileged users to the Protected Users security group. This special group was designed to harden privileged accounts and introduces a set of protection mechanisms, one of which is making NTLM authentication impossible for the members of this group. See https://learn.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/authentication-policies-and-authentication-policy-silos#about-authentication-policies for more details.

Have Breakglass Accounts in place

Break Glass accounts are emergency access accounts used to access critical systems or resources when other authentication mechanisms fail or are unavailable. In Active Directory, Break Glass accounts are used to provide emergency access to Active Directory in case normal T0 Admin accounts do not work anymore, e.g. because of a misconfigured Authentication Policy.

Clean Source Principle

The clean source principle requires all security dependencies to be as trustworthy as the object being secured. Implementation of the clean source principle is beyond the scope of this article, but explained in detail at Success criteria for privileged access strategy | Microsoft Learn.

Review ACLs on the Root-level of your Domain(s)

The security implications of an account having excessive privileges (e.g. being able to modify permissions at the root-level of the domain are massive. For that reason, before creating the new OU (named “Admin” in the description above), you must ensure that there are no excessive ACLs (Access Control List) configured on the Root-level of the domain. In addition to that, consider breaking inheritance on the OU called “Admin” in our example.

Updated Feb 19, 2024

Version 5.0
DagmarHeidecker