STAYING AHEAD OF RANSOMWARE, PART 2: PRIVILEGED ACCESS MANAGEMENT AND ZERO TRUST

What can we do to stop ransomware from happening in the first place? And how can we stay ahead of adversaries targeting the mainframe?

When thinking about ransomware prevention and privileged access, here are some helpful questions:

- How do you conduct change management?
- How do you track privileged access?
- What if someone has privileged access by default and their account is compromised?
- If you segment your network, why wouldn't you do the same for your mainframe?

Why is privileged access management (PAM) so important when it comes to ransomware? Just like on any other platform, once an attacker has initial access to the mainframe, they will often attempt to elevate their privileges. This will enable them to explore the environment, pivot laterally, and enumerate sensitive data and security settings with impunity.

We don't know what we don't manage, but what if we could effectively manage privileged access on the mainframe with a solution like BMC AMI Security?

During our customer penetration tests, we commonly discover accounts with excessive privileges or older, unused accounts with privileges. Remember: a key principle of zero trust (ZT) security is to never trust, always verify. There is no such thing as a known good entity or account. Instead, all access attempts and existing accounts are assumed to be potential attack vectors and must be verified and re-verified when requesting subsequent access. In addition, this access must be limited to the least amount of privilege required for that specific task, otherwise known as the principle of least privilege (PoLP).

This is a good segue to discuss the difference between PoLP and ZT. Though they are related, they are not the same.

Let's say I work at a factory on the assembly line. My badge will only allow me to enter the assembly line, because that's my designated job. I shouldn't have access to anywhere else in the factory, and if I attempt to do this, my badge won't allow it. PoLP is simple when applied to roles with minimal privileges.

Now let's say I'm the manager. My badge **should** just let me into the assembly line because PoLP dictates I have at least that amount of privilege as a manager. However, my access into this area still needs to be verified and audited by another party—every time. Why? Because maybe I'm not happy with the factory and the reason I want to enter the assembly line at 1 AM on a Sunday is to break some rivets. PoLP "should" let me into that area based on my privileges, but ZT requires that a third party is notified that I'm requesting access to the assembly line. So as a manager, I do have least privileges based on my role, but I really shouldn't be given any trust implicitly.

Side note: There has been discussion on whether it's constructive to tell employees, "We don't trust you." Rather than phrase it as, "We don't trust you," it may help to instead phrase ZT as, "We are doing this because we want to protect you and our entire organization." Or put differently: "We trust you. We just don't trust your credentials." This is 100 percent more reflective of the intentions behind ZT.

Also, the assembly line has cameras for monitoring (visibility/analytics) as part of a ZT model. In addition, when I scan my badge, there's another layer of authentication (multi-factor authentication, or MFA) that I must pass. All of these are ZT best practices.

The working assumption for all users in a ZT architecture is that they are granted the least amount of privilege (ideally none) as a baseline, as well as after verification. In other words, just because I am authenticated as a certain type of user, this does not imply that my privileges should also change as a result. Instead, my access rights and privileges will be the minimum possible to accomplish a given task.

So how does PAM help with tracking privileged access, enforcing PoLP, and implementing a ZT architecture?

A good PAM implementation must:

- Enumerate and categorize your privileged accounts. Any account that is not part of this established group will NOT have privileged access to the mainframe.
- Define more granular scopes of privilege rather than on an "all or nothing" basis. Just because an account is granted certain privileges, that does not mean it should have unfettered access to every application on the mainframe! Think of this as segmentation for the mainframe.
- Enforce PoLP for privileged accounts, and only when they need them! Define privileged access with limited privileges for specific periods of time.
- Audit access and activity of all privileged accounts.

Locking down privileged access on a PoLP basis and adhering to ZT fundamentals such as "never trust, always verify" will help you prevent a multitude of attacks before they can happen—including ransomware. That said, even with a mature PAM implementation, you must monitor for when an account elevates privileges without authorization. This would be a “smoking gun” alert related to privilege escalation and will help you respond in a timely manner.

In the next part of our series, we’ll discuss how **sensitive dataset monitoring** can help you stay ahead of the ransomware threat!

Check out part 1 of our series, Normalcy Bias and Initial Access