Finding Weak Passwords in Active Directory

blog.netwrix.com/2022/09/06/finding-weak-passwords-in-active-directory

Jeff Warren

Knowing the credentials for any user account in your network gives an adversary significant power. After logging on as a legitimate user, they can move laterally to other systems and escalate their privileges to deploy ransomware, steal critical data, disrupt vital operations and more.

Most organizations know this, and take steps to protect user credentials. In particular, they use Active Directory password policy to enforce password length, complexity and history requirements, and they establish a policy to lock out an account after a certain number of failed logon attempts. So they're safe, right?

Unfortunately not. Even with these controls in place, many people choose easily guessable passwords like *Winter2017* or *Password!@#* because they comply with company standards but are easy to remember. These weak passwords leave the organization vulnerable to one of the simplest attacks that adversaries use to gain a foothold in a network: guessing.

You might be surprised at just how well this strategy works. Let's walk through an example of a password guessing attack, and then explore how you can assess your vulnerability and strengthen your cybersecurity.

Handpicked related content:

Password Policy Best Practices for Strong Security in AD

How a password spraying attack works

In a password spraying attack, the adversary picks one commonly used password and tries using it to log on to each account in the organization. Most attempts will fail, but a single failed logon for an account will not trigger a lockout. If all the attempts fail, they simply try again with the next password in their arsenal. If they find a password that was chosen by just one user in your organization, they're inside your network, poised to wreak havoc.

One way an attacker can perform a password spraying attack is with CrackMapExec, a utility that's fee to download from Github. CrackMapExec comes bundled with a Mimikatz module (via PowerSploit) to assist with credential harvesting. Here's how the attack works:

Step 1. Check the Active Directory password policy and lockout policy.

To avoid lockouts, attackers need to know how many bad passwords they can guess per account. And to pick passwords that are likely to work, they need to know the company's AD password policy. CrackMapExec gives them both. Here is an example of the output it provides:

Now the attacker knows that in this environment, they have 9 guesses at each user's password without triggering a lockout. They can also see that the minimum password length is 5 characters and password complexity is enabled; this information can be used to craft a custom dictionary of candidate passwords without wasting guesses on passwords that would have been rejected by the policy. (Alternatively, they can use one of multiple password lists created using password dumps from data breaches, which are also readily available on GitHub.)

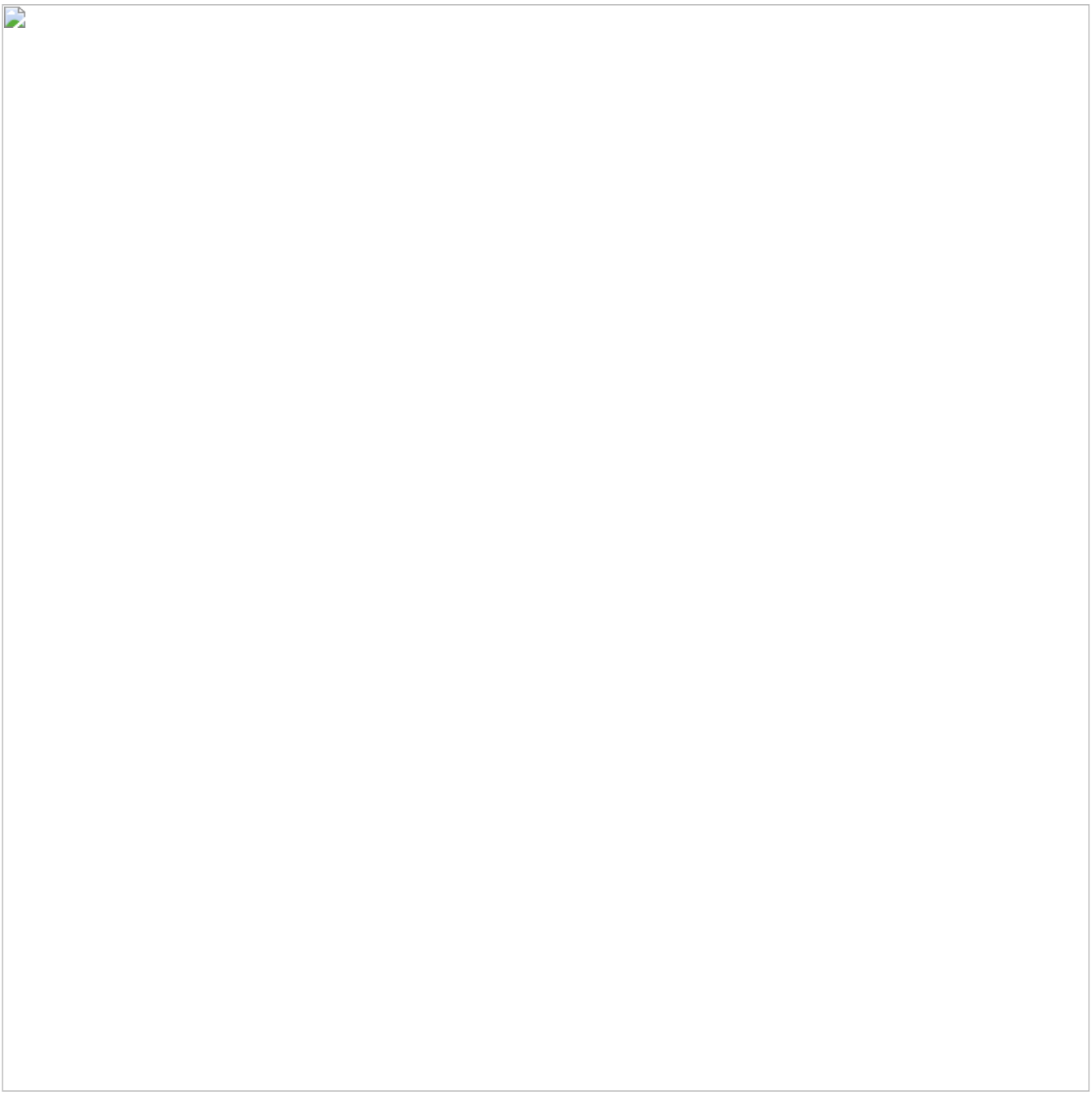
Step 2. Enumerate all user accounts.

Next, the adversary needs a list of accounts to try the passwords against. They can easily extract a list of all user accounts with an LDAP query, or they can use the **rid-brute** feature of CrackMapExec, as follows:

Step 3. Try each password against all user accounts.

With a list of all AD user accounts (users.txt) and a list of candidate passwords (passwords.txt), the adversary simply needs to issue the following command:

This command will try each password against each account until it finds a match:



Discovering your weak passwords

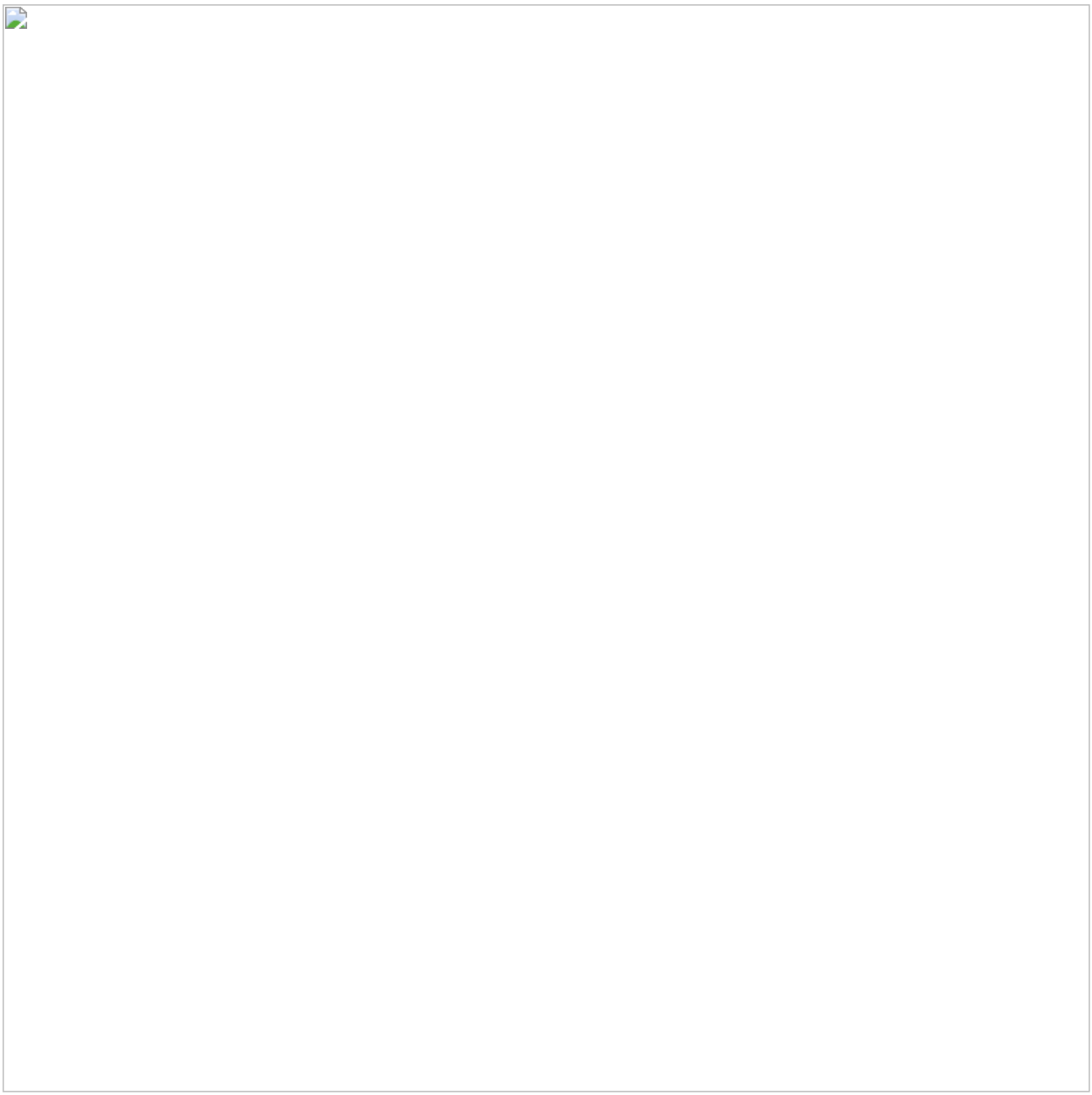
As you can see, attackers with no access rights in your environment have a very effective way to compromise your AD accounts: simply guessing their plaintext passwords. You may be wondering just how vulnerable your organization is to such attacks.

To find out, you can use the DSInternals command **Test-PasswordQuality**. It will extract the password hashes for all your user accounts and compare them against the password hashes for a dictionary of weak passwords.

Here is the command you can issue to run the analysis. It can be run remotely and will extract password hashes using DC replication similar to the DCSync Mimikatz attack.

At the top of the output report is a list of accounts stored with reversible encryption, a topic we covered in our last post.

Then the report lists all accounts whose passwords were found in the dictionary:



How Netwrix can help you defend against weak passwords

While Microsoft password policy enables you to put some constraints in place, it is not sufficient to prevent your users from choosing passwords that adversaries can easily guess. Netwrix offers an Active Directory security solution that enables you to require strong passwords. Even better, it enables you to secure your Active Directory from end to end. You can:

- Identify and mitigate vulnerabilities in your Active Directory, including not just weak passwords but excessive permissions, shadow admins, stale accounts and more.
- Enforce strong password policies and also control AD configurations and permissions to prevent credential theft.
- Detect even advanced threats to stop bad actors before they can complete their mission.
- Instantly contain a security breach with automated response actions, minimizing the damage to your business.
- Roll back or recover from malicious or otherwise improper changes with minimal downtime.

Jeff Warren