CyberArk Labs: From Safe Mode to Domain Compromise

By Doron Naim

Published: 2016-09-15 · Archived: 2026-04-06 01:10:38 UTC

Overview

CyberArk Labs recently identified what it believes to be a significant risk related to Windows Safe Mode, which is built into all Windows Operating Systems (OS) on both PCs and servers. Once attackers break through the perimeter and gain local administrator privileges on an infected Windows-based machine, they can remotely activate Safe Mode to bypass and manipulate endpoint security measures. In Safe Mode, the attackers are able to freely run tools to harvest credentials and laterally move to connected systems – all while remaining undetected. This exploit can also work in Windows 10, despite the presence of the Microsoft's Virtual Secure Module (VSM).

By exploiting these weaknesses, attackers can turn infected endpoints into launching points for pass-the-hash attacks, which can provide attackers with access to more machines on which they can re-use these same attack techniques to ultimately compromise the entire Windows environment.

Introduction

Safe Mode is a long-standing, basic function of the Windows OS that was designed to provide a lean environment in which users can examine and resolve issues that cannot be resolved in Normal Mode. Safe Mode was first released in 1995 when “security” and “cyber” were still terms of science fiction. Back then, vendors were focused on system stability, efficiency and business operations – meaning security was usually neglected. Yet, as the Internet has become ubiquitous, such technologies have become tools of opportunity for cyber attackers.

Because Safe Mode was purposely designed to be lean, it restricts most third-party software, including security tools, from running. As a result, cyber attackers on compromised machines can remotely reboot those machines into Safe Mode to disable and evade endpoint defenses and subsequently launch their attacks. Given the number of Windows systems in use, this risk impacts billions of PCs and servers globally.

This blog will outline the privileged attack pathway that starts with Safe Mode, address how Safe Mode can be used to manipulate endpoint security measures, and suggest steps your organization should take right now to mitigate this risk.

Description of Risk

It’s fairly easy for attackers to break through the perimeter and gain access to at least one machine on a corporate network. In fact, in a recent report by FireEye, 84 percent of organizations surveyed admitted to falling victim to at least one spear-phishing attack in 2015 (1). Once the attacker has access to a machine, the attacker will attempt to gain local administrator rights either by leveraging the user’s existing privileges or by using an exploit to elevate to these privileges.

Once attackers have remote, local administrator access to infected machines, they need to evade a variety of endpoint security measures such as anti-virus and endpoint threat detection tools. Next, to maximize their success, attackers typically scour endpoints looking for credentials that can be reused to laterally move throughout the network. With this in mind, several tools, including Microsoft’s recently released Virtual Secure Module (VSM), have been created to operate at the endpoint level to limit the use of attack tools and protect credentials from pass-the-hash attacks. These tools can be highly effective, but the catch is, they are typically designed to operate *only* in Normal Mode.

Enter, Safe Mode. Safe Mode, by design, does not boot any software or drivers that are not *critical to the operation of Windows*. As a result, by remotely forcing a reboot in Safe Mode, attackers inside a compromised system are able to operate freely, as most endpoint defenses are not enabled. And because VSM is only enabled in Normal Mode, attackers can also capture credential hashes needed to laterally move through the environment – despite Microsoft’s claims that pass-the-hash risks have been mitigated (2).

Exploiting Safe Mode to Escalate an Attack

This example explains how Safe Mode can be exploited to allow an attacker to capture user credentials and execute pass-the-hash attacks to achieve lateral movement. This pattern of credential capture and lateral movement can be reused by an attacker multiple times until an eventual domain compromise in achieved.

To successfully exploit this weakness in Safe Mode, attackers must complete three steps:

1. Change system settings to move the OS into Safe Mode during the next reboot
2. Configure attack tools to load in Safe Mode
3. Force the reboot of the machine to begin the exploit

This process is actually much easier than it sounds, and it can typically be done without the user noticing that anything has gone wrong. The steps below show just how simple this entire process can be.

Step 1: Remotely configure a machine to boot in Safe Mode

To remotely force a Windows-based machine into Safe Mode during the next reboot, attackers can use BCDEdit to configure the system to boot in Minimal Safe Mode. Once this change is made, the machine will – by default – boot in Minimal Safe Mode, which is the default Safe Mode boot option that runs only the minimal drivers and services needed to start Windows and prevents connections to the Internet and network.

Step 2: Configure attack tools to load in Safe Mode

Remember that, by design, Safe Mode loads only a minimal set of drivers and tools. To gain a presence in Safe Mode, the attacker must somehow enable his or her attack tools to run in this lean state. This can be done in a few ways, two of which include:

- **Malicious Service**. Attackers can create a malicious service that is configured to load in Safe Mode. The service can be included in the attacker's initial payload.
- **Malicious COM Object.** Attackers can register a malicious COM object that is loaded by explorer.exe. This enables that attacker's code to run each time the explorer.exe needs to parse icons. (Yes, surprisingly this works in Safe Mode, as well.)

With these tools in place, the attacker's malicious code will automatically run during the next reboot.

Since most endpoint security solutions are not effective in Minimal Safe Mode, these attack tools can easily evade endpoint security measures. In this state, the attacker is able to freely use his or her tools to steal credentials from LSASS.exe and then reuse those credentials to continue the attack path of lateral movement and privilege escalation.

Step 3: Restart the machine to execute the next phase of the attack

This can easily be done in a variety of ways, including directly from the command line in Normal Mode.

But wait! How does the attacker do this without the victim noticing? Sure, the attacker can arbitrarily force a restart, but this will likely look suspicious to the user and prompt a phone call to the IT team. Instead, to stay under the radar, the attacker can also either wait until the next restart or show the victim an "update" window with a message that says the PC must be rebooted. This "update" window can purposely be designed to look like a legitimate Windows pop-up.

Next, depending on the attacker's goal, there are a few techniques an attacker can use to continue to stay hidden from the victim. Let's look at the techniques based on the attacker's goal:

- **Credential Theft**. If the attacker's goal is to steal credentials for future use, then the attacker actually *wants* the user to log on to the system. As the user logs in, the attacker can capture the credentials. In this case, the attacker will likely use the COM object technique to execute code that will change the background, look and feel of Safe Mode – making it *appear* that the user is still in Normal Mode. As soon as the user enters his or her credentials, a second "update" window can prompt the user to reboot yet again to move the machine back into the actual Normal Mode. Just as mentioned above, this secondary reboot prompt can mimic a legitimate Windows prompt to prevent the user from noticing anything suspicious.
- **Lateral Movement**. If the attacker's goal is to perform a pass-the-hash attack using previously compromised credentials, then the attacker does not need the user to login. In this case, the attacker is better off creating a service. At the time of reboot, the service can automatically run code to execute a pass-the-hash attack and then immediately reboot the machine again back into Normal Mode. These back-to-back restarts are indistinguishable to the user, and thus further prevent the user from noticing that something went wrong. Based on tests conducted by CyberArk Labs, we found this technique to be highly effective in stealthily enabling lateral movement.

What about Event Log? Someone can definitely see that a PC went to Safe Mode. That is correct, but at this point I'll encourage you read to research by my colleague, Roi Cohen, "Can Incident Response and Audit Teams Always Trust Windows Security Event Logs?"

Exploiting Safe Mode to Manipulate Security Measures

The next example will show how attackers can not only avoid endpoint security defenses in Safe Mode but also manipulate these solutions within Safe Mode. This example was tested against McAfee LiveSafe, Avira Free Antivirus, Trend Micro Maximum Security 10 and Windows Defender.

All of these solutions run in Normal Mode and some can run in Network Safe Mode, which is a Safe Mode boot option that boots the minimal drivers and services need to start Windows, as well as the drivers and services needed connect to the Internet and network. However, none of these solutions runs in Minimal Safe Mode. As such, registry keys associated with these solutions cannot be modified in Normal Mode or Network Safe Mode, but they *can* be modified in Minimal Safe Mode.

Once an attacker has booted a machine into Minimal Safe Mode, the attacker can access registry keys and alter configurations to disable or manipulate endpoint security solutions. Once that's complete, an attacker can reboot the machine back into Normal Mode and freely proceed with the attack without the risk of being blocked by an endpoint security solution.

When testing this in our lab, we attempted to use Mimikatz in Normal Mode to steal credentials from memory. When endpoint security solutions were properly configured, Mimikatz was removed each time we attempted to use it – including when we attempted to copy it to the disk. After we modified the registry keys in Minimal Safe Mode, we were able to seamlessly run Mimikatz, write the tool to the disk, and successfully capture credentials without setting off any alarms.

Mitigation

Though this risk is built in to all Windows OSes, there are a few proactive steps organizations can take to mitigate this risk. Recommendations include the following:

- **Enforce the principle of least privilege.** The ability to remotely load a machine into Safe Mode from Normal Mode is only present when an attacker is able to operate with local administrator privileges. By removing local administrator privileges from standard users, organizations can reduce their exposure to this type of exploit.
- **Rotate privileged account credentials.** To successfully execute a pass-the-hash attack, the compromised credential hash must still be valid. By proactively rotating privileged account credentials, organizations can frequently and automatically invalidate password hashes that may exist on machines throughout the environment. As a result, attackers can still capture the password hashes from compromised machines, but the hashes will be useless.
- **Employ security tools that operate in Safe Mode.** Security tools that only function properly in Normal Mode can leave organizations exposed to this type of attack. When evaluating endpoint security tools, ensure that the solution you select is able to function in Safe Mode.
- **Monitor the use of Safe Mode.** This can be done both proactively and after the fact. Set alerts to know when a machine boots in Safe Mode, and monitor the Windows Event Log to search for this type of event. Note, however, that recent research indicates that the Windows Event Log may not be entirely trusted.

Disclosure Timeline

- February 24, 2016: Initial discovery by CyberArk Labs
- May 2, 2016: Risk reported to Microsoft Security Response Center
- May 2, 2016: Microsoft responded that they did not consider the submission a valid vulnerability as it requires an attacker to have already compromised the machine.*
- September 15, 2016: Public disclosure

*According to Microsoft's "Ten Immutable Laws of Security"

1 https://www2.fireeye.com/rs/fireye/images/fireeye-how-stop-spearphishing.pdf

2 https://www.blackhat.com/docs/us-15/materials/us-15-Moore-Defeating%20Pass-the-Hash-Separation-Of-Powers.pdf

---