Threat hunting for PsExec and other lateral movement tools

By Tony Lambert

Archived: 2026-04-05 16:26:12 UTC

Adversaries in post-compromise security incidents are like shoppers in a grocery store. It's rare that they head straight to a single place to get what they need. Instead, they enter the building and move around the aisles, gathering pieces along the way before they check out. In the same way, adversaries won't hit a single place on your network that contains all the data they're trying to steal. They have to move laterally from system to system (or aisle to aisle) to gather the information that they came for.

To move around freely without attracting too much attention, attackers often use reliable software that looks normal in an enterprise environment. One of the favorites among adversaries is an old friend of many IT administrators: SysInternals PsExec.

In this post, we're going to discuss some tactics that threat hunters can use to identify instances where adversaries use PsExec (even when it's been renamed or cloned) and similar tools to move laterally between endpoints on your network.

The Classic: What is PsExec?

SysInternals first released PsExec more than a decade ago, and it has provided administrators a reliable method for remotely accessing systems via the Server Message Block (SMB) protocol ever since. The fundamental behavior of PsExec follows a simple pattern:

1. Establishes an SMB network connection to a target system using administrator credentials
2. Pushes a copy of a receiver process named PSEXESVC.EXE to the target system's ADMIN$ share
3. Launches PSEXESVC.EXE, which sends input and output to a named pipe

In general, a named pipe is a method of interprocess communication, and various specific pipes are common in Windows Active Directory domains. Pipes may be named for specific uses, and, in this case, a pipe for PsExec communication usually looks like this: `\\.\pipe\psexesvc.` This detail becomes incredibly important when searching for malicious uses of PsExec in your environment because even an evasive, renamed version of PsExec will still use named pipes to communicate. In fact, this behavior is so predictable that we even see it in other software that merely clones the functionality of PsExec. (We'll come back to this in a moment).

Binary metadata also helps identify renamed instances of PsExec. For the source process establishing a connection, PsExec's internal name value is simply `PsExec`, while the target's receiving process, PsExeSvc.exe, has an internal name of `PsExec Service Host.` Since PsExec is primarily available as precompiled binaries, these metadata aren't easily changed and can be handy to determine execution.

In addition to the metadata, a Windows Registry artifact (such as `HKEY_CURRENT_USER\software\sysinternals\psexec\eulaaccepted`) can indicate when PsExec has been used.

Additionally, an admin must accept an End-User License Agreement if they want to use the utility, and the acceptance is written to this key. As with binary metadata and named pipe usage, this will not change if PsExec is renamed, but it can be easily removed by an informed adversary.

Send in the Clones

Since PsExec is easy to use and reliable, it became a natural choice for legit software and adversaries alike, as each needed a way of issuing commands to remote systems. PsExec's licensure terms, however, do not allow for redistribution within other software packages, which presented a problem for software developers, so now there are a variety of open-source tools that clone the capabilities of PsExec.

RemCom

RemCom is an open-source, redistributable utility providing the same remote management functions. It achieved a level of notoriety after adversaries used it to move laterally in their attack on the Democratic National Committee in 2016. However, it's also included in several legitimate software packages. By default, RemCom sends `RemComSvc.exe` to a remote computer, which then uses the named pipe `\\.\pipe\remcom_comunication` (misspelling and all) in the place of PsExec's named pipe. In addition, the process's internal name has a value of `remcom`.

PAExec

PAExec features all the same functions of RemCom and PsExec, and is primarily intended for use with the PowerAdmin server management solution. By default, PAExec uses a named pipe containing the string `PAExec` combined with a unique process identifier and computer name values. The receiving process is similarly named and usually has an internal name value of `paexec.`

CSExec

CSExec is a highly configurable, C# implementation of PsExec's functionality. By default, CSExec sends `csexecsvc.exe` to the remote computer and uses a named pipe called `\\.\pipe\csexecsvc.` The internal name value is readily changeable—although this depends on compilation instructions—but it should contain the string `csexec` by default either way.

Comparing and contrasting all of these clones, a few things should become apparent:

- All the clones are open-source, with their source-code available to the public
- Adversaries can download the code and recompile it with different values for named pipe names and binary internal names
- Despite this, the underlying functionality for communication will remain the same

Hunting for Clones

Now that we've established a reliable behavior for an entire class of lateral movement tools, we can devise a detection strategy. Detection based on named pipes can get a bit messy because pipes are commonplace on Windows endpoints, but we can get a head start by excluding the pipes that we know are benign. Within Active Directory environments, you can expect to find named pipes such as:

- `\\.\pipe\netlogon`
- `\\.\pipe\samr`
- `\\.\pipe\lsarpc`

You can also find other legitimate ones in specialized environments for things like SQL database queries and in mainframe systems. In your environment, you can establish a baseline of named pipes by using Sysinternals PipeList or Sysmon with Windows Event Logging. If you leverage endpoint detection and response (EDR) tools in your environment, then you can examine process file modifications to find named pipes as well.

Once you've found your baseline, you can start searching for abnormal pipes. Normal pipes should be abundant, while abnormal ones should be uncommon.

What Else Can You Find With Pipes?

With the data we just collected about normal named pipes, you can also hunt for additional lateral movement and reconnaissance in your environment. For example, tools like Cobalt Strike and Metasploit both support lateral movement using named pipes. In the case of Cobalt Strike, a default pipe name containing the string "msagent" is common, but this can be changed easily.

In the case of Metasploit, the adversary must specify a pipe name during configuration instead of accepting a default. In both cases, looking for outlier pipes in a dataset can reveal the usage of these tools, although confirmation generally requires further investigation within your specific environment. This is especially handy as payloads from Cobalt Strike and Metasploit can be injected into other legitimate processes to evade detection. We've found that a unique named pipe can make an awesome indicator of compromise during an incident.

On the reconnaissance side, you can detect tools like Bloodhound or Sharphound performing Active Directory enumeration. These tools communicate with domain infrastructure over ordinary named pipes that are common across Windows systems, but they do so in extreme excess. Analyzing a named pipe dataset for a large amount of named pipe communications originating from a single process on a single host can lead you to find the origin of this enumeration. Just like Cobalt Strike and Metasploit payloads, Bloodhound can be injected into legitimate processes or executed using PowerShell in unexpected ways. As such, detection through pipes can help narrow down the suspect activity.

Pipes Are Noisy, How Can I Get Better Results?

After you gather a dataset of named pipes from your endpoints, you'll probably notice that there are a lot of processes using a lot of pipes. One way to decrease this noise is to limit the use of specific processes so that the outliers can stand out more. For example, if you want to increase the fidelity of hunts for PsExec in your organization, then you could ban its use by administrators in favor of PowerShell Remoting. PowerShell Remoting can perform the same actions as PsExec—and it does so more securely. Making little changes like this can help you slowly improve your hunt results while teaching you more about your environment!

Wait, What About the Binary Metadata?

Don't fret if you can't gather information about named pipes at scale in your environment right now. If you have access to binary metadata in your environment, then you can start searching for suspicious lateral movement using these searches:

- Binary internal name is `psexec` or `Psexec Service Host`, but the process name isn't `psexec.exe` or `psexesvc.exe`
- Binary internal name is `remcom` but the process name isn't `remcom.exe` or `RemComSvc.exe`
- Binary internal name includes `paexec` but the process name doesn't contain `paexec`
- Binary internal name includes `csexec` but the process name doesn't contain `csexec`

These searches are a good starting point if you're looking for renamed instances of these tools, but they may prove ineffective against certain open-source variants, especially those that have been custom-compiled by more determined adversaries.

Conclusion

Despite the wealth of lateral movement tools similar to PsExec, an evergreen tactic for detecting these tools is to hunt for outlying named pipes used by processes on your Windows endpoints. Hunting in this manner will help reveal activity from unauthorized remote management tools in multiple forms, even when the tools have been modified by adversaries. Collecting and analyzing named pipe data will also have benefits outside the realm of PsExec clones, allowing you to find adversary tools for covert movement and reconnaissance that would otherwise be difficult to find.

If you're interested in further reading, I wrote about some methods for detecting lateral movement when adversaries use Windows Remote Management (WinRM) and Windows Management Instrumentation (WMI) almost exactly one year ago.

> To take a deeper dive into threat hunting for lateral movement and other ATT&CK tactics, watch our Lateral Movement webinar series.
