The Nearest Neighbor Attack: How A Russian APT Weaponized Nearby Wi-Fi Networks for Covert Access

By mindgrub

Published: 2024-11-22 · Archived: 2026-04-02 12:41:05 UTC

In early February 2022, notably just ahead of the Russian invasion of Ukraine, Volexity made a discovery that led to one of the most fascinating and complex incident investigations Volexity had ever worked. The investigation began when an alert from a custom detection signature Volexity had deployed at a customer site ("Organization A") indicated a threat actor had compromised a server on the customer's network. While Volexity quickly investigated the threat activity, more questions were raised than answers due to a very motivated and skilled advanced persistent threat (APT) actor, who was using a novel attack vector Volexity had not previously encountered. At the end of the investigation, Volexity would tie the breach to a Russian threat actor it tracks as GruesomeLarch (publicly known as APT28, Forest Blizzard, Sofacy, Fancy Bear, among other names). Volexity further determined that GruesomeLarch was actively targeting Organization A in order to collect data from individuals with expertise on and projects actively involving Ukraine.

The month-and-a-half long investigation revealed that GruesomeLarch was able to ultimately breach Organization A's network by connecting to their enterprise Wi-Fi network. The threat actor accomplished this by daisy-chaining their approach to compromise multiple organizations *in close proximity* to their intended target, Organization A. This was done by a threat actor who was thousands of miles away and an ocean apart from the victim. Volexity is unaware of any terminology describing this style of attack and has dubbed it the **Nearest Neighbor Attack**.

The Nearest Neighbor Attack

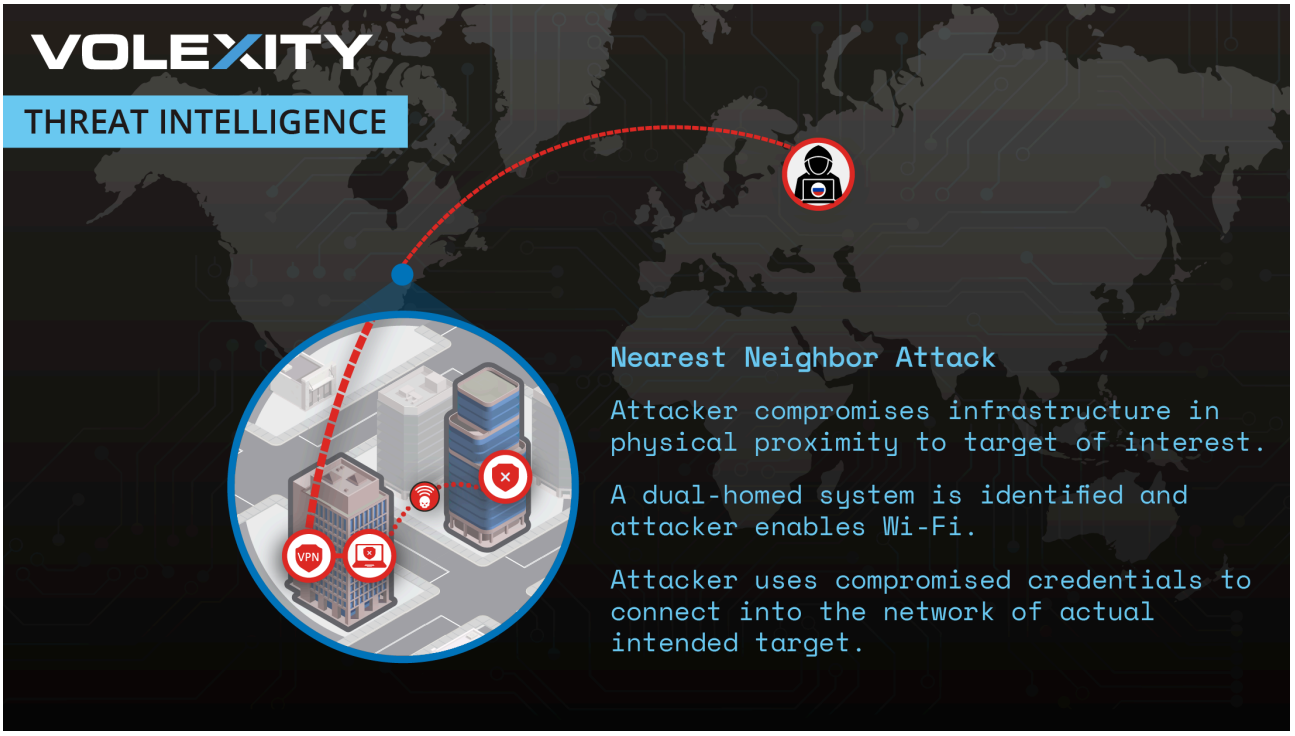
Given the assumed physical distance, you may be wondering how exactly this threat actor was able to authenticate to Organization A's Enterprise Wi-Fi network. The first and most obvious thing needed were valid credentials. This was accomplished via password-spray attacks against a public-facing service on Organization A's network to validate credentials. And while credentials could be validated, they could not be used against Organization A's public services due to implementation of multi-factor authentication (MFA).

The Enterprise Wi-Fi network, however, did not require MFA and only required a user's valid domain username and password to authenticate. Meanwhile, the threat actor was halfway around the world and could not actually connect to Organization A's Enterprise Wi-Fi network. To overcome this hurdle, the threat actor worked to compromise other organizations who were in buildings within close proximity to Organization A's office. Their strategy was to breach another organization, and then move laterally within that organization to find systems they could access that were dual-homed, (i.e., having both a wired and wireless network connection).

Once successful in this endeavor, having found a system that was connected to the network via a wired Ethernet connection, the threat actor would access the system and use its Wi-Fi adapter. At this point they would connect to

the SSID of Organization A's Enterprise Wi-Fi and authenticate to it, thus granting them access to Organization A's network.

The anatomy of the Nearest Neighbor Attack is shown below.



At this point it would be understandable if you're thinking this sounds a little far-fetched. However, Volexity discovered that GruesomeLarch was successful in breaching *more than one* organization within close proximity to Organization A. And they were able to find and compromise a dual-homed system at a nearby organization, allowing them to connect to Organization A's Enterprise Wi-Fi network from that compromised system.

Volexity believes this represents a new class of attack that has not previously been described, in which a threat actor compromises one organization and performs credential-stuffing attacks in order to compromise other organizations in close physical proximity via their Wi-Fi networks. To reiterate, the compromise of these credentials alone did not yield access to the customer's environment, as all Internet-facing resources required use of multi-factor authentication (MFA). However, the Wi-Fi network was not protected by MFA, meaning proximity to the target network and valid credentials were the only requirements to connect.

This blog post aims to shed light on the tactics, techniques, and procedures (TTPs) Volexity observed during its incident investigation, and to provide a detailed look at how the Nearest Neighbor Attack worked and ways to mitigate against it.

A Glimpse of a Ghost

This story begins on February 4, 2022, when Volexity detected suspicious activity on the network of its customer, Organization A. This detection came after the triggering of an alert from a custom signature Volexity had developed to look for files being written to and executed out of the root of the C:\ProgramData directory. As Volexity investigated, it could see the following activity had occurred:

- A file named C:\ProgramData\servtask.bat had been written and executed.
- A file named servtask.bat had invoked the Microsoft command-line registry utility and PowerShell to run the following commands:
    - reg save hklm\sam C:\ProgramData\sam.save
    - reg save hklm\security C:\ProgramData\security.save
    - reg save hklm\system C:\ProgramData\system.save
    - Powershell -c “Get-ChildItem C:\ProgramData\sam.save, C:\ProgramData\security.save, C:\ProgramData\system.save ^| Compress-Archive -DestinationPath C:\Program\Dataout.zip”

This immediately put the Volexity threat detection & response team on high alert, as they could see sensitive registry hives were being exported and compressed into a ZIP file. The next steps in the investigation were now clear: the team immediately expanded their deep dive into the system’s EDR event history and prepared a package to deploy Volexity Surge Collect Pro to collect system memory (RAM) and key disk artifacts.

The review of the EDR logs provided some interesting insight into activities that immediately preceded and came shortly after the initial finding. Volexity found the following had occurred on the system just ahead of the activity mentioned above:

- A login with an unprivileged user account on the server occurred over RDP.
- A file named DefragmentSrv.zip appeared on the system under that user’s directory and was unarchived using the GUI version of WinRAR that was on the system.
- Two files, DefragmentSrv.exe and DefragmentSrv.bat, were also written and executed; that chain ultimately led to the writing and execution of servtask.bat.
- A file named wayzgoose52.dll was also written to a bogus directory located at C:\ProgramDataA\dobev3.80.15456.

Volexity was keen to collect these files as part of the incident response investigation. Unfortunately, the team encountered two major issues. The first issue was that, in the middle of collecting system memory, the system was shut down. This was less than ideal, as it resulted in the loss of volatile data that may have proven useful to the investigation. It is possible components of those files would have still been memory resident and allowed their recovery and analysis. Despite this setback, Volexity was able to have the server powered back on and still collected forensics artifacts to support the investigation. However, the second issue was that Volexity found the attacker had removed all the files and folders that had been identified. Not only were the files gone, but Volexity discovered the attacker had run a native Microsoft utility called `Cipher.exe`, which covered their tracks by securely erasing the files. While this is not the first time Volexity has encountered an attacker covering their tracks with anti-forensics methods, it is the first time Volexity had seen it done with the Cipher.exe utility.

Investigation Dead Ends

After the initial incident, the attacker laid low for a while. In the meantime, Volexity used various forensic artifacts it collected to continue its investigation. Some challenges were encountered in the process. First, Volexity was able to identify an IP address that had connected to the victim server, but it was not immediately clear what it was related to, and it was no longer online. Further, Volexity had a network security sensor deployed in Organization

A's office, but there was virtually no record of the attacker on it. A few steps that would often immediately provide clarity and next steps in an investigation were frustratingly coming up empty.

The investigation went cold for a period of time before the attacker resurfaced. When the attacker returned, Volexity was able to get *some* answers. Volexity learned the IP address segment the attacker was coming from was associated with Organization A's Enterprise Wi-Fi network, and one of the domain controllers on the network acted as a DHCP server. However, after examining the DHCP logs, there was no record of the IP addresses Volexity had tied to the attacker. Another possible lead was another dead end.

Wireless Redemption

A bit further into the investigation, Volexity learned the organization had a wireless controller that was used to manage its wireless network, access points, and all related infrastructure. This controller also kept detailed logs related to signal strength, connected devices, authenticated user accounts, and so on. Volexity obtained access to the wireless controller and had its first major break in the investigation. Not long after getting access to the wireless controller, Volexity was able to find the IP address of the attacker and tie it to an authenticated domain user and a MAC address.

Armed with this new information, Volexity was able to examine Organization A's RADIUS logs and find authentication events tied to the user and MAC address that had just been discovered. This same MAC address and user account appeared in older logs overlapping with the initial breach. However, Volexity found additional authentication events with the same MAC address and a different username going back to late January 2022. Volexity learned the password for the account observed from January had expired and been updated by the user. This apparently locked the attacker out of that account, but they were able to come back in early February with the account observed from the wireless controller.

This new information caused Volexity to examine logs from a system at Organization A that provided Internet-facing webservices that could be authenticated against. The service itself was protected with MFA, but it could be used to verify if credentials were valid or not. Upon examining those logs, Volexity found that in January and February, password-spray attacks had been carried out against this service and three accounts had been successfully compromised by an attacker. Two of the three accounts identified were those Volexity had identified from the wireless controller and RADIUS logs. The third account had not yet been used.

Volexity now determined the attacker was connecting to the network via wireless credentials they had brute-forced from an Internet-facing service. However, it was not clear where the attacker was physically that allowed them to connect to the Enterprise Wi-Fi to begin with. Further analysis of data available from Organization A's wireless controller showed which specific wireless access points the attacker was connecting to and overlayed them on a map that had a layout of the building and specific floors. Volexity could see the attacker was connecting to the same three wireless access points that were in a conference room at the far end of the building near windows along the street. This gave Volexity the first evidence that the "*call was not coming from inside the building.*" Could this be an attacker conducting a close access operation from the street outside? Nothing was ruled out, but Volexity was not too far off from discovering the real answer.

Trust No One, Especially Your Neighbor

During the course of this investigation, Volexity worked with Organization A to take various remediation steps, implement countermeasures, and make improvements to various areas where logging and network visibility were not optimal. Doing this ultimately allowed Volexity to make a huge break in the investigation and figure out exactly what was going on.

Despite resetting various credentials, including the three accounts identified from the password-spray attacks, the attacker still had working domain credentials. The attacker once again regained access to the Organization A's Enterprise Wi-Fi, but with visibility and logging now in place, Volexity quickly jumped into action. Volexity was now able to pull full packet capture from all activity involving Wi-Fi connected systems in Organization A's network, regardless of where they connected internally. Analysis of this packet capture revealed the attacker's system had sent out NetBIOS Name Service (NBNS) queries that revealed its computer name and the active directory domain to which it was joined. This active directory domain revealed exactly where the attacker was connecting from, which turned out to be an organization ("Organization B") located *right across the street*.

Volexity was able to get in touch with Organization B and work with them to investigate this matter further. This is where Volexity ultimately uncovered how the attacker was operating, and how the Nearest Neighbor Attack worked. In coordination with Organization B, Volexity was able to find the system that had connected to Organization A's Wi-Fi. Analysis of this system showed it had been breached after an attacker used privileged credentials to connect to it via RDP from another system within Organization B's network. This system was dual-homed and connected to the Internet via wired Ethernet, but it also had a Wi-Fi network adapter that could be used at the same time. The attacker found this system and used a custom PowerShell script to examine the available networks within range of its wireless, and then connected to Organization A's Enterprise Wi-Fi using credentials they had compromised. A redacted copy of the C# code embedded in the custom PowerShell script is available here.

Additional analysis of systems at Organization B revealed the intruder had two modes of access to their network. The first was with credentials that allowed them to connect to their VPN, which was not protected with MFA. Volexity also found evidence the attacker had been connecting to Organization B's Wi-Fi from another network that belonged to another nearby organization ("Organization C"). The attacker had gone to great lengths to breach multiple organizations so they could daisy-chain Wi-Fi and/or VPN connections to ultimately reach the network of Organization A. The team at Volexity was astounded but also breathed a sigh of relief that it now had an explanation and evidence of how this attack was happening.

Volexity was able to determine who Organization C was based on analysis of MAC addresses and SSID information and contacted them. However, Organization C opted not to provide Volexity with access to key data required to take the investigation further. In any case, all of these findings gave Volexity a full understanding of the attacker's operations and allowed the team to confidently recommend further mitigations and remediation instructions to Organization A. At this point the attacker's access was cut off from Organization A's Enterprise Wi-Fi, and they have not been observed connecting to this network since then.

One Final Hurrah: The Guest Wi-Fi

Over a month after the last observed threat actor activity, and following various remediation steps, Volexity had yet another alert indicating suspect activity in the network of its customer, Organization A. Upon investigating that

activity, Volexity discovered the same threat actor had managed to return to the network and was proxying through multiple internal systems. Fortunately, Volexity was able to rapidly investigate this incident and walk it back through multiple systems to its origin: a system on Organization A's Guest Wi-Fi network.

While the Guest Wi-Fi network had been believed to be completely isolated from the corporate wired network, where the high-value targeted data resided, there was one system that was accessible from both the Wi-Fi network and the corporate wired network. Armed with the credentials of an account that had not been reset, and the fact that the Wi-Fi network was not completely isolated, the attacker was able to pivot back into the corporate wired network and ultimately regain access to the high-value targeted data.

To achieve this pivot, the attacker used the Windows utility `netsh` to set up a series of port-forwards that allowed them to reach the target systems. Example commands used for this are provided below:

```
cmd.exe /C netsh advfirewall firewall add rule name="Remote Event Log Management SMB"
dir=in action=allow protocol=tcp localport=12345 > C:\Windows\Temp\MSI28122Ac.LOG 2>&1

cmd.exe /C netsh interface portproxy add v4tov4 listenaddress=172.33.xx.xx listenport=12345
connectaddress=172.20.xx.xx connectport=445 > C:\Windows\Temp\MSI2cBfA24.LOG 2>&1
```

The source system connecting in to instigate this activity could once again be determined through analysis of network traffic and logs from Organization A's wireless controller. Volexity found that the attacker was connecting in this time from Organization C. Volexity again contacted Organization C and also worked with Organization A to take new remediation steps to resolve this new intrusion.

Since this final activity related to the Guest Wi-Fi network, there has been no observed activity that Volexity can tie to an attacker leveraging the Nearest Neighbor Attack.

Tradecraft Notes

GruesomeLarch predominantly adopted a living-off-the-land approach during their intrusion, leveraging standard Microsoft protocols and moving laterally. The sections that follow detail some observations from the incident that could be used to potentially detect GruesomeLarch or other threat actors using similar behaviors or techniques.

Use of Cipher.exe

During the intrusion, the attacker removed files they created, making use of an inbuilt Windows tool, `Cipher.exe`, that ships with every modern version of Windows:

```
λ cipher /?
Displays or alters the encryption of directories [files] on NTFS partitions.

  CIPHER [/E | /D | /C]
         [/S:directory] [/B] [/H] [pathname [...]]

  CIPHER /K [/ECC:256|384|521]

  CIPHER /R:filename [/SMARTCARD] [/ECC:256|384|521]

  CIPHER /P:filename.cer

  CIPHER /U [/N]
```

The following functionality was used to overwrite deleted data in a particular folder:

> `cmd.exe /c cipher /W:C`

The Microsoft documentation describes this in the following way:

> To overwrite deleted data on a volume by using Cipher.exe, use the `/w` switch with the cipher command:
>
> 1. Quit all programs.
> 2. Select **Start** > **Run**, type *cmd*, and then press `Enter`.
> 3. Type `cipher /w:<directory>`, and then press ENTER, where `<directory>` is any folder in the volume that you want to clean. For example, the `cipher /w:C` command causes all deallocated space on drive C to be overwritten. If `<directory>` is a mount point or points to a folder on another volume, all deallocated space on that volume will be cleaned.

The effect is that attackers are able to securely delete their tools using native Windows functionality without bringing a new tool or writing their own code, thus making recovery of attacker tools more difficult for forensic analysts.

The attacker in this case was meticulous in their use of this tool, with every file that Volexity identified as having been written to disk being later deleted by this tool. It is worth noting that in its numerous incident response engagements, Volexity has not previously identified an attacker cleaning up after themselves using this technique.

Dumping Ntds.dit via VSSAdmin

Another observed tactic was the attempt to steal the active directory database by creating a volume shadow copy. This is a common technique and one that Volexity has seen for several years. The workflow is well documented publicly and consists of the following key components that were seen in this incident:

- Create a volume shadow copy, e.g., the following:

> `vssadmin create shadow /for C: /quiet`

- Retrieve a copy of the `ntds.dit` file and the SYSTEM registry hive from the volume shadow copy:

> `copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDSNTDS.dit [dest]`

```
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\System32\config\SYSTEM [dest]
```

- Download the copied files. To download the files (which were fairly large) the attacker compressed them using a PowerShell command:

```
powershell -c "& { Add-Type -Assembly 'System.IO.Compression.FileSystem';
[IO.Compression.ZipFile]::CreateFromDirectory($path1', '$path2');}" >
C:\Windows\Temp\b2rMBPL.tmp 2>&1
```

Antivirus and endpoint detection and response (EDR) products may naturally detect this behavior as being potentially malicious. However, for additional detection opportunities, organizations can create custom EDR signatures to look for a privileged account which exhibits the following:

- Any use of vssadmin.exe
- Copying or moving of files from the VolumeShadowCopy directories
- PowerShell commands indicating in-line compression of files

Staging Data for Exfiltration

The majority of the data from this incident was copied back to the attacker's system, which was connected to the Wi-Fi. However, in a few cases, Volexity observed the attacker staging data in directories on a public-facing webserver. These files were then exfiltrated via external downloads.

This is a common technique that Volexity sees attackers use in a variety of breaches. It can be difficult to monitor for this activity, but there are opportunities to detect this behavior if organizations can monitor for unexpected files on webservers or large file transfers that are out of the ordinary. If nothing else, ensuring web logs are working and being saved can assist with investigations later on.

Attribution

Initially, Volexity was not able to attribute this intrusion to a known threat actor. The attacker was largely using living-off-the-land techniques, and any tooling or IP addresses they used made it difficult for Volexity to zero in on a possible culprit. However, once Volexity was able to determine who and what was being targeted internally, it immediately suspected that this was the activity of a Russian threat actor, but which one?

Then, in April 2024, Microsoft published research on *Forest Blizzard*, which Volexity tracks as GruesomeLarch, detailing a post-compromise tool named GooseEgg that the threat actor had used. This tool was leveraged in the zero-day exploitation of CVE-2022-38028, a privilege escalation vulnerability in the Microsoft Windows Print Spooler service. In their report, Microsoft detailed several key file names, folder paths, and commands used by the framework, notably the following:

- Servtask.bat
- Wayzgoose52.dll
- DefragmentSrv.exe
- C:\ProgramData\[var]\v%u.%02u.%04u

These exact file names and paths were observed in the incident investigated by Volexity. Microsoft's report also showed what commands were in the `servtask.bat` file, which were identical to what Volexity had seen where registry hives had been saved and compressed into a file named `out.zip` from the initial intrusion activity. However, as documented in the Tradecraft Notes section of this blog post, these files had been securely deleted using the `Cipher.exe` tool.

Microsoft's post stated that GooseEgg had been in use since "*at least June 2020 and possibly as early as April 2019.*" Volexity can confirm this tool was definitively used in February 2022. Exploitation of CVE-2022-38028 also provides an explanation as to how the initial victim system was likely compromised. Based on the use of this tool, which Microsoft indicates is unique to this threat actor, Volexity assesses with high confidence that the activity described in this post can be attributed to GruesomeLarch.

Conclusion

Volexity's investigation reveals the lengths a creative, resourceful, and motivated threat actor is willing to go to in order to achieve their cyber-espionage objectives. The Nearest Neighbor Attack effectively amounts to a close access operation, but the risk of being physically identified or detained has been removed. This attack has all the benefits of being in close physical proximity to the target, while allowing the operator to be thousands of miles away.

Organizations need to place additional considerations on the risks that Wi-Fi networks may pose to their operational security. A significant amount of effort over the last several years has been placed on attack surface reduction where Internet-facing services have been secured with MFA or removed altogether. However, the same level of care has not necessarily been given to Wi-Fi networks. It may be time to treat access to corporate Wi-Fi networks with the same care and attention that other remote access services, such as virtual private networks (VPNs), have received.

This attack was possible due to a lower level of security controls on targeted Wi-Fi systems than other resources, such as email or VPN. In this case, an attacker figured out how to abuse these controls, even though they were far beyond their geographic reach, using the following workflow:

- Compromise an organization in the physical geographic vicinity of their target.
- Find a dual-homed system in that network, which has both Ethernet and Wi-Fi connectivity.
- Examine the available Wi-Fi networks within reach of the compromised, dual-homed system being accessed.
- Brute-force credentials for organizations related to those Wi-Fi networks.
- Authenticate to physically adjacent Wi-Fi networks using discovered credentials.

Using the Nearest Neighbor Attack method, the attacker was able to daisy-chain their way from organization to organization without ever deploying malware, using only valid user credentials as their access method. The attacker then focused on using living-of-the-land techniques to avoid deploying malware and to evade detection by EDR products.

To generally prevent or detect attacks similar to those discussed in this blog, Volexity recommends the following:

- Monitor and alert on anomalous use of the netsh and Cipher.exe utilities within your environment.
- Create custom detection rules to look for files executing from various non-standard locations, such as the root of C:\ProgramData\.
- Detect and identify exfiltration of data from Internet-facing services run in your environment.
- Create separate networking environments for Wi-Fi and Ethernet-wired networks, particularly where Ethernet-based networks allow for access to sensitive resources.
- Consider hardening access requirements for Wi-Fi networks, such as applying MFA requirements for authentication or certificate-based solutions.
- Monitor network traffic between devices to identify files being transferred via SMB that contain commonly exfiltrated data (credential data, ntds.dit, registry hives, etc.).

Acknowledgement

Volexity would like to thank its customer for granting permission to publicly share information about this investigation.
