Frozen in transit: Secret Blizzard's AiTM campaign against diplomats | Microsoft Security Blog

By Microsoft Threat Intelligence

Published: 2025-07-31 · Archived: 2026-04-05 17:43:24 UTC

Microsoft Threat Intelligence has uncovered a cyberespionage campaign by the Russian state actor we track as Secret Blizzard that has been targeting embassies located in Moscow using an adversary-in-the-middle (AiTM) position to deploy their custom ApolloShadow malware. ApolloShadow has the capability to install a trusted root certificate to trick devices into trusting malicious actor-controlled sites, enabling Secret Blizzard to maintain persistence on diplomatic devices, likely for intelligence collection. This campaign, which has been ongoing since at least 2024, poses a high risk to foreign embassies, diplomatic entities, and other sensitive organizations operating in Moscow, particularly to those entities who rely on local internet providers.

While we previously assessed with low confidence that the actor conducts cyberespionage activities within Russian borders against foreign and domestic entities, this is the first time we can confirm that they have the capability to do so at the Internet Service Provider (ISP) level. This means that diplomatic personnel using local ISP or telecommunications services in Russia are highly likely targets of Secret Blizzard's AiTM position within those services. In our previous blog, we reported the actor likely leverages Russia's domestic intercept systems such as the System for Operative Investigative Activities (SORM), which we assess may be integral in facilitating the actor's current AiTM activity, judging from the large-scale nature of these operations.

This blog provides guidance on how organizations can protect against Secret Blizzard's AiTM ApolloShadow campaign, including forcing or routing all traffic through an encrypted tunnel to a trusted network or using an alternative provider—such as a satellite-based connection—hosted within a country that does not control or influence the provider's infrastructure. The blog also provides additional information on network defense, such as recommendations, indicators of compromise (IOCs), and detection details.

Secret Blizzard is attributed by the United States Cybersecurity and Infrastructure Agency (CISA) as Russian Federal Security Service (Center 16). Secret Blizzard further overlaps with threat actors tracked by other security vendors by names such as VENOMOUS BEAR, Uroburos, Snake, Blue Python, Turla, Wraith, ATG26, and Waterbug.

As part of our continuous monitoring, analysis, and reporting of the threat landscape, we are sharing our observations on Secret Blizzard's latest activity to raise awareness of this actor's tradecraft and educate organizations on how to harden their attack surface against this and similar activity. Although this activity poses a high risk to entities within Russia, the defense measures included in this blog are broadly applicable and can help organizations in any region reduce their risk from similar threats. Microsoft is also tracking other groups using similar techniques, including those documented by ESET in a previous publication.

AiTM and ApolloShadow deployment

In February 2025, Microsoft Threat Intelligence observed Secret Blizzard conducting a cyberespionage campaign against foreign embassies located in Moscow, Russia, using an AiTM position to deploy the ApolloShadow malware to maintain persistence and collect intelligence from diplomatic entities. An adversary-in-the-middle technique is when an adversary positions themself between two or more networks to support follow-on activity. The Secret Blizzard AiTM position is likely facilitated by lawful intercept and notably includes the installation of root certificates under the guise of Kaspersky Anti-Virus (AV). We assess this allows for TLS/SSL stripping from the Secret Blizzard AiTM position, rendering the majority of the target's browsing in clear text including the delivery of certain tokens and credentials. Secret Blizzard has exhibited similar techniques in past cyberespionage campaigns to infect foreign ministries in Eastern Europe by tricking users to download a trojanized Flash installer from an AiTM position.

Initial access

In this most recent campaign, the initial access mechanism used by Secret Blizzard is facilitated by an AiTM position at the ISP/Telco level inside Russia, in which the actor redirects target devices by putting them behind a captive portal. Captive portals are legitimate web pages designed to manage network access, such as those encountered when connecting to the internet at a hotel or airport. Once behind a captive portal, the Windows Test Connectivity Status Indicator is initiated—a legitimate service that determines whether a device has internet access by sending an HTTP GET request to *hxxp://www.msftconnecttest[.]com/redirect* which should direct to *msn[.]com*.

Delivery and installation

Once the system opens the browser window to this address, the system is redirected to a separate actor-controlled domain that likely displays a certificate validation error which prompts the target to download and execute ApolloShadow. Following execution, ApolloShadow checks for the privilege level of the *ProcessToken* and if the device is not running on default administrative settings, then the malware displays the user access control (UAC) pop-up window to prompt the user to install certificates with the file name *CertificateDB.exe*, which masquerades as a Kaspersky installer to install root certificates and allow the actor to gain elevated privileges in the system.

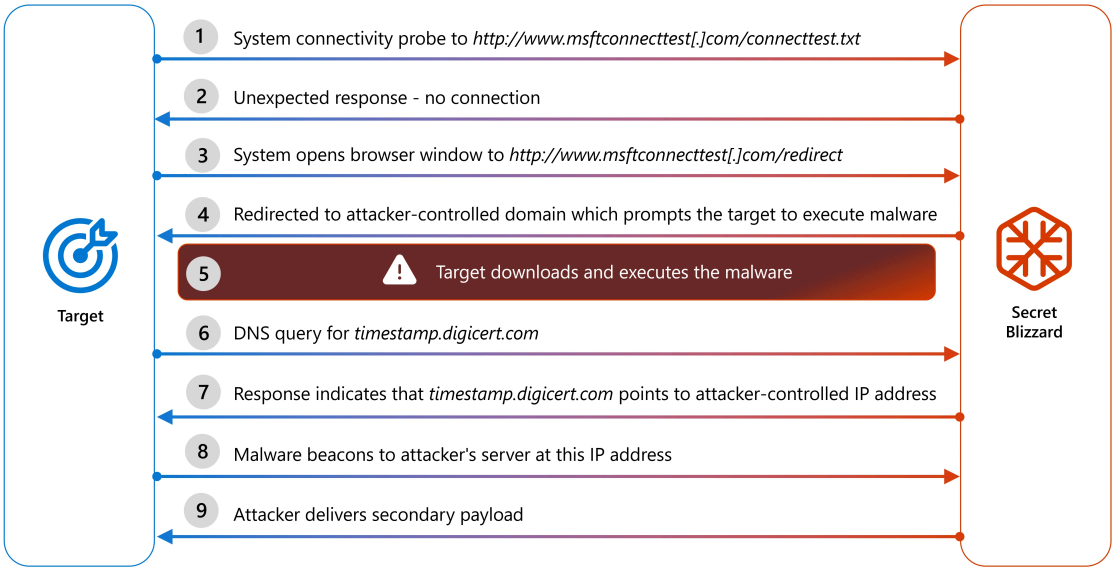


Figure 1. Secret Blizzard AiTM infection chain

ApolloShadow malware

ApolloShadow uses two execution paths depending on the privilege level of the running process. The token of the running process is retrieved using the API *GetTokenInformationType* and the value of *TokenInformation* is checked to see if the token contains the *TokenElevationTypeFull*type**.** If it does not have that privilege level, ApolloShadow executes a low privilege execution path**.**

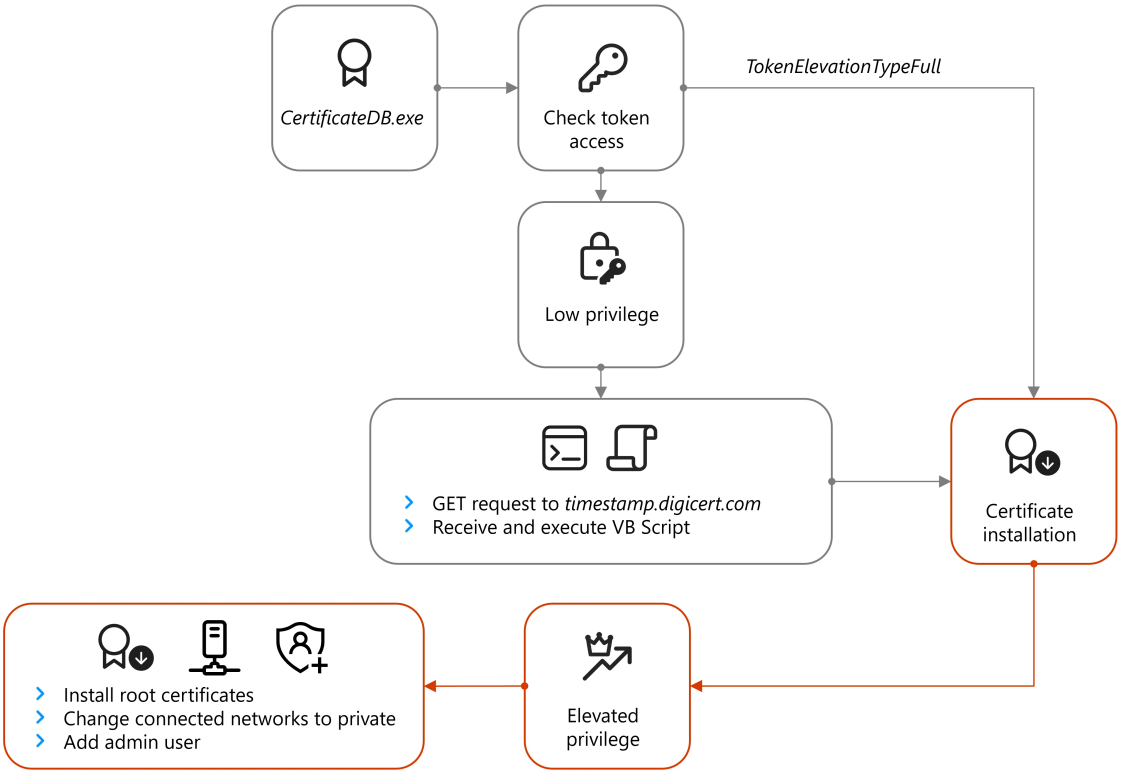


Figure 2. ApolloShadow execution flow

Low privilege execution

When executing the low privilege path, the first action is to collect information about the host to send back to the AiTM controlled command and control (C2). First, the host's IP information is collected using the API *GetIpAddrTable*, which collects information from the *IpAddrTable*. Each entry is individually Base64-encoded and delineated by a pipe character with *\r\n* appended, then combined into one string. For example:

- 172.29.162[.]128 00-15-5D-04-04-1C
- 127.0.0[.]1

```
"|MTcyLjI5LjE2Mi4xMjggMDAtMTUtNUQtMDQtMDQtMUM=|\r\n|MTI3LjAuMC4xIA==|\r\n"
```

Then the entire string is Base64-encoded once again in preparation for exfiltration to the C2 host:

```
"fE1UY3lMakk1TGpFMk1pNHhNamdnTURBdE1UVXROVVF0TURRdE1EUXRNVU09fA0KfE1USTNMakF1TUM0eElBPT18DQo="
```

The encoded network information is added as a query string to a GET request with the destination URL *hxxp://timestamp.digicert[.]com/registered*. Two query parameters are included with the request, *code* and *t*. The *Code* parameters contains a hardcoded set of characters and the *t* variable has the encoded IP address information, as shown below:

```
code=DQBBBBBBBBBOBBBBBBBBBBgBBBBBBBBBny_t???????
t=fE1UY3lMakk1TGpFMk1pNHhNamdnTURBdE1UVXROVVF0TURRdE1EUXRNVU09fA0KfE1USTNMakF1TUM0eElBPT18DQo=
```

While the timestamp subdomain does exist for Digicert, the /*registered* resource does not. Due to the AiTM position of the actor, Secret Blizzard can use DNS manipulation to redirect legitimate-looking communication to the actor-controlled C2 and return an encoded VBScript as the second-stage payload.

When the response comes back from the redirected Digicert request, the file name that is used to write the script to disk is decoded for use. ApolloShadow uses string obfuscation in several places throughout the binary to hide critical strings. These strings are blocks of encoded characters that are encoded using XOR with a separate set of hardcoded constants. While this is not a particularly sophisticated technique, it is enough to obscure the strings from view at first glance. The strings are decoded as they are used and then re-encoded after use to remove traces of the strings from memory.

```
vbs_filename ^= 0x77F817A23BAE3B03uLL;
qword_7FF6C216F288 ^= 0xD738388F74839CB8uLL;
v34 = 0;
v35 = 0;
v36 = 0;
tmpPathLen = -1;
v4 = -1;
do
  ++v4;
while ( *((_BYTE *)&vbs_filename + v4) );
E_copyStr(&v34, &vbs_filename, v4);
vbs_filename ^= 0x77F817A23BAE3B03uLL;
qword_7FF6C216F288 ^= 0xD738388F74839CB8uLL;
```

Figure 2. String decoding operation for VB script name

The decoded file name is *edgB4ACD.vbs* and the file name string is concatenated by the malware with the results of querying the environment variable for the *TEMP* directory to create the path for the target script. We were unable to recover the script, but the header of the response is checked for the first 12 characters to see if it matches the string *MDERPWSAB64B*. Once ApolloShadow has properly decoded the script, it executes the script using the Windows API call *CreateProcessW* with the command line to launch *wscript* and the path to *edgB4ACD.vbs*.

Finally, the ApolloShadow process launches itself again using *ShellExecuteA*, which presents the user with an UAC window to bypass UAC mechanisms and prompt the user to grant the malware the highest privileges

available to the user.

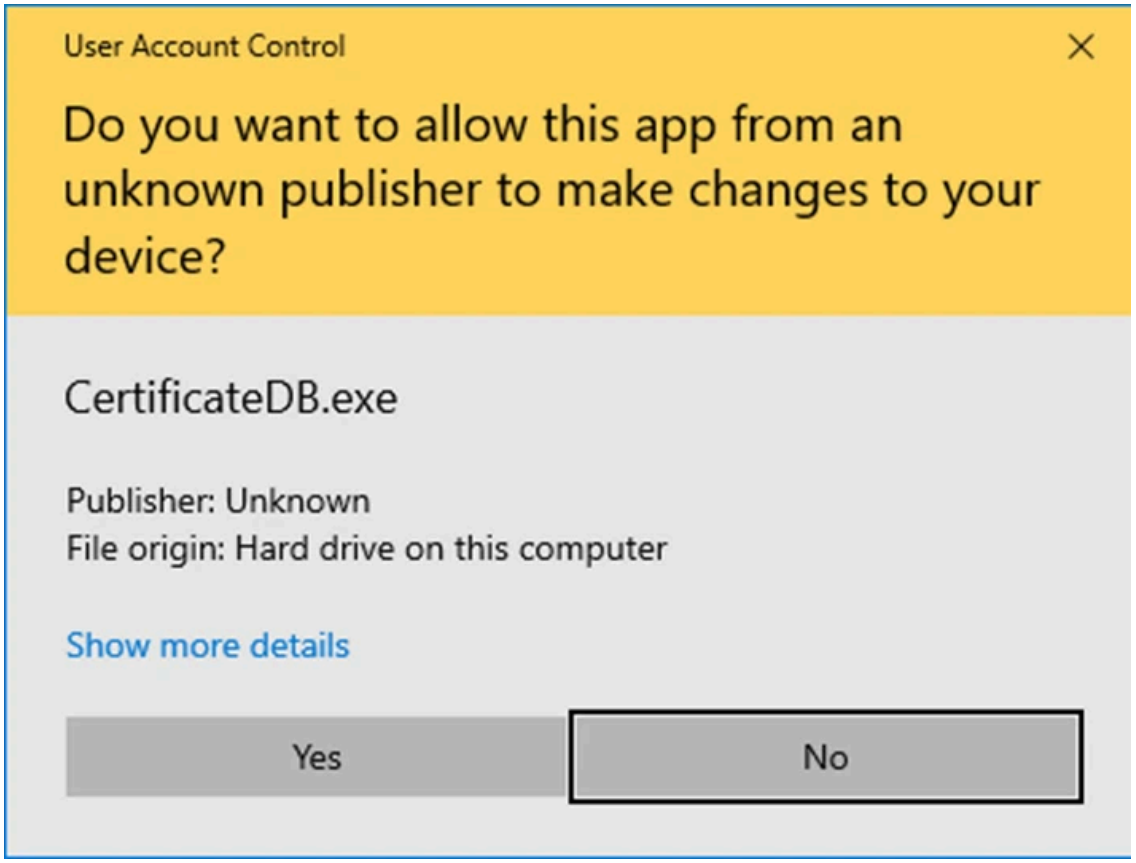
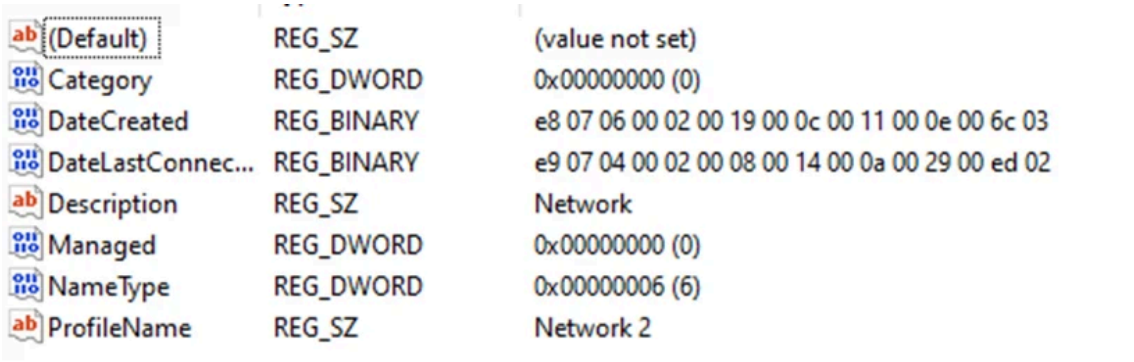


Figure 3. UAC popup to request elevated privileges from the user

Elevated privilege execution

When the process is executed with sufficient elevated privileges, ApolloShadow alters the host by setting all networks to *Private*. This induces several changes including allowing the host device to become discoverable, and relaxing firewall rules to enable file sharing. While we did not see any direct attempts for lateral movement, the main reason for these modifications is likely to reduce the difficulty of lateral movement on the network. ApolloShadow uses two different methods to perform this change.

The first method is through the registry settings for *NetworkProfiles: SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkList\\Profiles*. The network's globally unique identifiers (GUIDs) are parsed for each connected network, and the malware modifies the value *Category* by setting it to 0. This change sets the profile of the network to *Private* after the host has been rebooted.



(Default)	REG_SZ	(value not set)
Category	REG_DWORD	0x00000000 (0)
DateCreated	REG_BINARY	e8 07 06 00 02 00 19 00 0c 00 11 00 0e 00 6c 03
DateLastConnec...	REG_BINARY	e9 07 04 00 02 00 08 00 14 00 0a 00 29 00 ed 02
Description	REG_SZ	Network
Managed	REG_DWORD	0x00000000 (0)
NameType	REG_DWORD	0x00000006 (6)
ProfileName	REG_SZ	Network 2

Figure 4. Registry settings for network profiles

The second method directly sets firewall rules using Component Object Model (COM) objects that enable file sharing and turn on network discovery. Several strings are decoded using the same method as above and concatenated to create the firewall rules they want to modify.

- `FirewallAPI.dll,-32752`
  - This command enables the **Network Discovery** rule group
- `FirewallAPI.dll,-28502`
  - This command enables all rules in the **File and Printer Sharing** group

The strings are passed to the COM objects to enable the rules if they are not already enabled.

```
if ( *FW_Rule )
{
  v1 = CoInitializeEx(0, COINIT_APARTMENTTHREADED);
  if ( ((int)(v1 + 0x80000000) < 0 || v1 == (unsigned int)RPC_E_CHANGED_MODE)
    && CoCreateInstance(&HNetCfg_FwPolicy2, 0, 1u, &CLSID_INetFwPolicy2, (LPVOID *)&This) >= 0
    && This->lpVtbl->get_Rules(This, &rules) >= 0
    && CoCreateInstance(&HNetCfg_FwRule, 0, 1u, &CLSID_INetFwRule, (LPVOID *)&v4) >= 0
    && v4->lpVtbl->put_Grouping(v4, *FW_Rule) >= 0
    && ((int (__fastcall *)(INetFwRule *, __int64))v4->lpVtbl->put_Profiles)(v4, 2) >= 0
    && ((int (__fastcall *)(INetFwPolicy2 *, __int64, _QWORD, __int16 *))This->lpVtbl->IsRuleGroupEnabled)(
         This,
         2,
         *FW_Rule,
         &v6) >= 0
    && !v6 )
  {
    ((void (__fastcall *)(INetFwPolicy2 *, __int64, _QWORD, __int64))This->lpVtbl->EnableRuleGroup)(
      This,
      0x7FFFFFFF,
      *FW_Rule,
      1);
  }
}
SysFreeString(*FW_Rule);
```

Figure 5. COM objects used to modify firewall rules

Both techniques have some crossover, but the following table provides a comparison overview of each method.

Technique	Purpose	Timing	Stealth	Effect
Registry profile change	Sets network to *Private*	Requires reboot	High	Broadly relaxes firewall posture

COM-based rule enablement	Activates specific rules	Immediate	Moderate	Opens precise ports for discovery and sharing

From here, ApolloShadow presents the user with a window showing that the certificates are being installed.

Figure 6. Window displayed to the user during execution

A new thread performs the remainder of the functionality. The two root certificates being installed are written to the *%TEMP%* directory with a temporary name and the extension *crt*. The certificate installation is performed by using the Windows certutil utility and the temporary files are deleted following the execution of the commands.

- `certutil.exe -f -Enterprise -addstore root "C:\Users\<username>\AppData\Local\Temp\crt3C5C.tmp"`
- `certutil.exe -f -Enterprise -addstore ca "C:\Users\<username>\AppData\Local\Temp\crt53FF.tmp"`

The malware must add a preference file to the Firefox preference directory because Firefox uses different certificate stores than browsers such as Chromium, which results in Firefox not trusting the root and enterprise store by default. ApolloShadow reads the registry key that points to the installation of the application and builds a path to the preference directory from there. A file is written to disk called *wincert.js* containing a preference modification for Firefox browsers, allowing Firefox to trust the root certificates added to the operating system's certificate store.

- `pref("security.enterprise_roots.enabled", true);" privilege`

The final step is to create an administrative user with the username *UpdatusUser* and a hardcoded password on the infected system using the Windows API *NetUserAdd*. The password is also set to never expire.

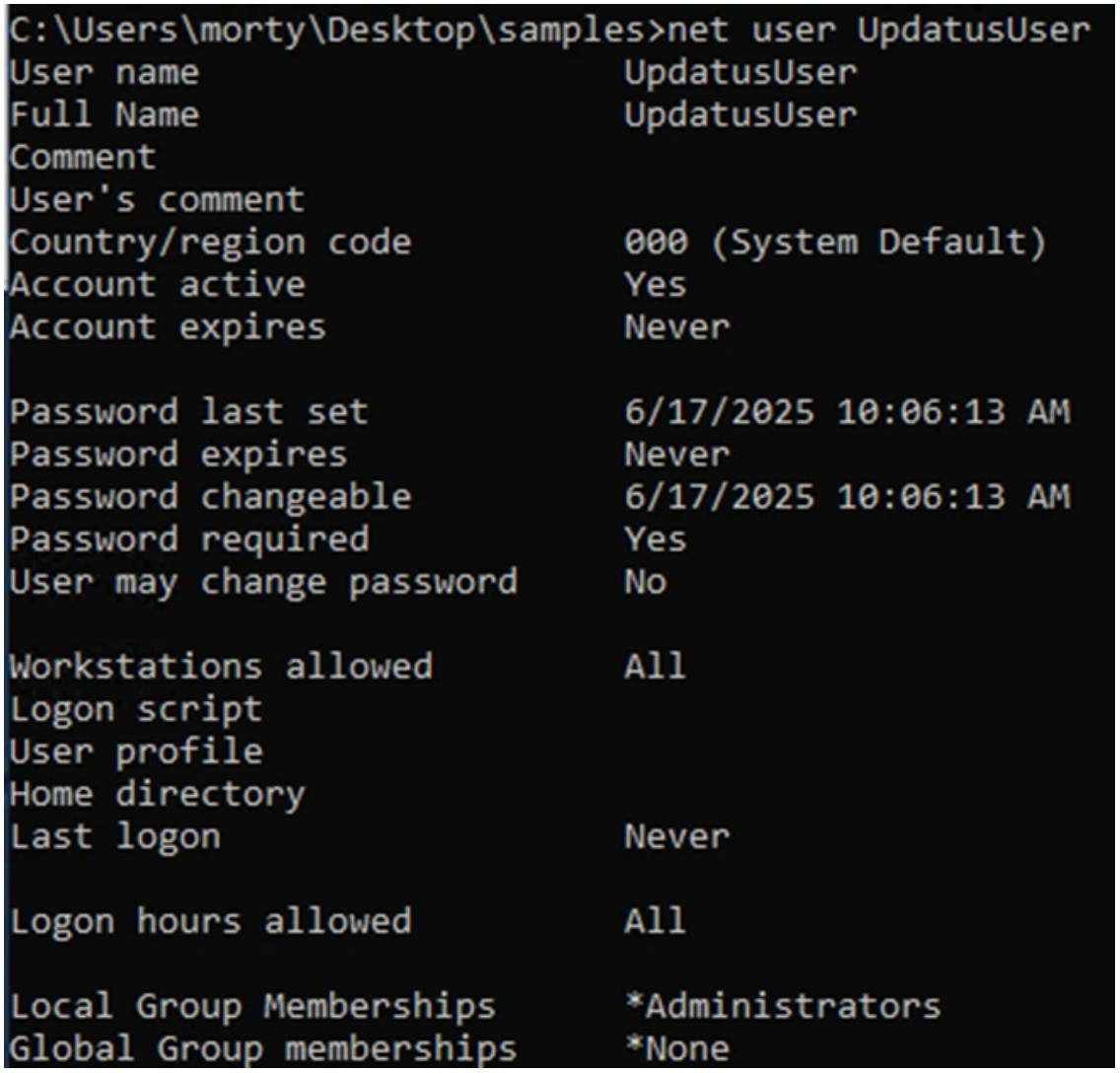


Figure 7. Administrator user added to infected system

ApolloShadow has successfully installed itself on the infected host and has persistent access using the new local administrator user.

Defending against Secret Blizzard activity

Microsoft recommends that all customers, but especially sensitive organizations operating in Moscow, should implement the following recommendations to mitigate against Secret Blizzard activity.

- Route all traffic through an encrypted tunnel to a trusted network or use a virtual private network (VPN) service provider, such as a satellite-based provider, whose infrastructure is not controlled or influenced by outside parties.

Microsoft also recommends the following guidance to enhance protection and mitigate potential threats:

- Practice the principle of least privilege, use multifactor authentication (MFA), and audit privileged account activity in your environments to slow and stop attackers. Avoid the use of domain-wide, admin-level service accounts and restrict local administrative privileges. These mitigation steps reduce the paths that

attackers have available to them to accomplish their goals and lower the risk of the compromise spreading in your environment.

- Regularly review highly privileged groups like Administrators, Remote Desktop Users, and Enterprise Admins. Threat actors may add accounts to these groups to maintain persistence and disguise their activity.
- Turn on cloud-delivered protection in Microsoft Defender Antivirus or the equivalent for your antivirus product to cover rapidly evolving attacker tools and techniques.
- Run endpoint detection and response (EDR) in block mode, so that Defender for Endpoint can block malicious artifacts, even when your non-Microsoft antivirus doesn't detect the threat or when Microsoft Defender Antivirus is running in passive mode. EDR in block mode works behind the scenes to remediate malicious artifacts detected post-breach.
- Turn on attack surface reduction rules to prevent common attack techniques. These rules, which can be configured by all Microsoft Defender Antivirus customers and not just those using the EDR solution, offer significant hardening against common attack vectors.
- Block executable files from running unless they meet a prevalence, age, or trusted list criterion
- Block execution of potentially obfuscated scripts

Microsoft Defender XDR detections

Microsoft Defender XDR customers can refer to the list of applicable detections below. Microsoft Defender XDR coordinates detection, prevention, investigation, and response across endpoints, identities, email, apps to provide integrated protection against attacks like the threat discussed in this blog.

Customers with provisioned access can also use Microsoft Security Copilot in Microsoft Defender to investigate and respond to incidents, hunt for threats, and protect their organization with relevant threat intelligence.

Microsoft Defender Antivirus

Microsoft Defender Antivirus detects this threat as the following malware:

- Trojan:Win64/ApolloShadow

Microsoft Defender for Endpoint

The following alerts might indicate threat activity related to this threat. Note, however, that these alerts can be also triggered by unrelated threat activity.

- Secret Blizzard Actor activity detected
- Suspicious root certificate installation
- Suspicious certutil activity
- User account created under suspicious circumstances
- A script with suspicious content was observed

Microsoft Security Copilot

Security Copilot customers can use the standalone experience to create their own prompts or run the following pre-built promptbooks to automate incident response or investigation tasks related to this threat:

- Incident investigation
- Microsoft User analysis
- Threat actor profile
- Threat Intelligence 360 report based on MDTI article
- Vulnerability impact assessment

Note that some promptbooks require access to plugins for Microsoft products such as Microsoft Defender XDR or Microsoft Sentinel.

Threat intelligence reports

Microsoft customers can use the following reports in Microsoft products to get the most up-to-date information about the threat actor, malicious activity, and techniques discussed in this blog. These reports provide the intelligence, protection information, and recommended actions to prevent, mitigate, or respond to associated threats found in customer environments.

Microsoft Defender Threat Intelligence

- Actor profile: Secret Blizzard

Microsoft Security Copilot customers can also use the Microsoft Security Copilot integration in Microsoft Defender Threat Intelligence, either in the Security Copilot standalone portal or in the embedded experience in the Microsoft Defender portal to get more information about this threat actor.

Hunting queries

Microsoft Defender XDR

Microsoft Defender XDR customers can run the following query to find related activity in their networks:

Surface devices that attempt to download a file within two minutes after captive portal redirection. This activity may indicate a first stage AiTM attack—such as the one utilized by Secret Blizzard—against a device.

```
let CaptiveRedirectEvents = DeviceNetworkEvents
| where RemoteUrl contains "msftconnecttest.com/redirect"
| project DeviceId, RedirectTimestamp = Timestamp, RemoteUrl;
let FileDownloadEvents = DeviceFileEvents
| where ActionType == "FileDownloaded"
| project DeviceId, DownloadTimestamp = Timestamp, FileName, FolderPath; CaptiveRedirectEvents
```

```
| join kind=inner (FileDownloadEvents) on DeviceId
| where DownloadTimestamp between (RedirectTimestamp .. (RedirectTimestamp + 2m))
| project DeviceId, RedirectTimestamp, RemoteUrl, DownloadTimestamp, FileName, FolderPath
```

Microsoft Sentinel

Microsoft Sentinel customers can use the TI Mapping analytics (a series of analytics all prefixed with ‘TI map’) to automatically match the malicious domain indicators mentioned in this blog post with data in their workspace. If the TI Map analytics are not currently deployed, customers can install the Threat Intelligence solution from the Microsoft Sentinel Content Hub to have the analytics rule deployed in their Sentinel workspace.

Below are the queries using Sentinel Advanced Security Information Model (ASIM) functions to hunt threats across both Microsoft first party and third-party data sources. ASIM also supports deploying parsers to specific workspaces from GitHub, using an ARM template or manually.

Detect network IP and domain indicators of compromise using ASIM

The below query checks IP addresses and domain indicators of compromise (IOCs) across data sources supported by ASIM Network session parser.

```
//IP list and domain list- _Im_NetworkSession
let lookback = 30d;
let ioc_ip_addr = dynamic(["45.61.149.109"]);
let ioc_domains = dynamic(["kav-certificates.info"]);
_Im_NetworkSession(starttime=todatetime(ago(lookback)), endtime=now())
| where DstIpAddr in (ioc_ip_addr) or DstDomain has_any (ioc_domains)
| summarize imNWS_mintime=min(TimeGenerated), imNWS_maxtime=max(TimeGenerated),
    EventCount=count() by SrcIpAddr, DstIpAddr, DstDomain, Dvc, EventProduct, EventVendor
```

Detect network and files hashes indicators of compromise using ASIM

The below queries will check IP addresses and file hash IOCs across data sources supported by ASIM Web session parser.

Detect network indicators of compromise and domains using ASIM

```
//IP list - _Im_WebSession
let lookback = 30d;
let ioc_ip_addr = dynamic(["45.61.149.109"]);
```

```
let ioc_sha_hashes =dynamic(["13fafb1ae2d5de024e68f2e2fc820bc79ef0690c40dbfd70246bcc394c52ea20"]);
_Im_WebSession(starttime=todatetime(ago(lookback)), endtime=now())
| where DstIpAddr in (ioc_ip_addr) or FileSHA256 in (ioc_sha_hashes)
| summarize imWS_mintime=min(TimeGenerated), imWS_maxtime=max(TimeGenerated),
    EventCount=count() by SrcIpAddr, DstIpAddr, Url, Dvc, EventProduct, EventVendor
// Domain list - _Im_WebSession
let ioc_domains = dynamic(["kav-certificates.info"]);
_Im_WebSession (url_has_any = ioc_domains)
```

Detect files hashes indicators of compromise using ASIM

The below query will check IP addresses and file hash IOCs across data sources supported by ASIM FileEvent parser.

Detect network and files hashes indicators of compromise using ASIM

```
// file hash list - imFileEvent
let ioc_sha_hashes =dynamic(["13fafb1ae2d5de024e68f2e2fc820bc79ef0690c40dbfd70246bcc394c52ea20"]);
imFileEvent
| where SrcFileSHA256 in (ioc_sha_hashes) or
TargetFileSHA256 in (ioc_sha_hashes)
| extend AccountName = tostring(split(User, @'')[1]),
    AccountNTDomain = tostring(split(User, @'')[0])
| extend AlgorithmType = "SHA256"
```

Indicators of compromise

Indicator	Type	Description
kav-certificates[.]info	Domain	Actor-controlled domain that downloads the malware

45.61.149[.]109	IP address	Actor-controlled IP address
13fafb1ae2d5de024e68f2e2fc820bc79ef0690c40dbfd70246bcc394c52ea20	SHA256	ApolloShadow malware
e94c00fde5bf749ae6db980eff492859d22cacb4bc941ad4ad047dca26fd5616	SHA256	ApolloShadow malware
CertificateDB.exe	File name	File name associated with ApolloShadow sample

References

- https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-129a
- https://www.welivesecurity.com/2018/01/09/turlas-backdoor-laced-flash-player-installer/
- https://attack.mitre.org/techniques/T1557/
- https://web-assets.esetstatic.com/wls/2018/01/ESET_Turla_Mosquito.pdf

Acknowledgments

- https://securelist.com/compfun-successor-reductor/93633/

Learn more

Meet the experts behind Microsoft Threat Intelligence, Incident Response, and the Microsoft Security Response Center at our VIP Mixer at Black Hat 2025. Discover how our end-to-end platform can help you strengthen resilience and elevate your security posture.

For the latest security research from the Microsoft Threat Intelligence community, check out the Microsoft Threat Intelligence Blog.

To get notified about new publications and to join discussions on social media, follow us on LinkedIn, X (formerly Twitter), and Bluesky.

To hear stories and insights from the Microsoft Threat Intelligence community about the ever-evolving threat landscape, listen to the Microsoft Threat Intelligence podcast.
