Pulsedive Blog | Pikabot Rising

By Pulsedive Threat Research

Published: 2024-01-22 · Archived: 2026-04-06 01:26:49 UTC

Pikabot is an emerging loader that has been active since early 2023. The malware provides access to environments with the ability to remotely execute commands for reconnaissance or the ingress of additional tools. Over the past year, security researchers have observed Pikabot distributed through malspam and malvertising. Its usage became more prevalent following the takedown of Qakbot by law enforcement agencies in 2023.

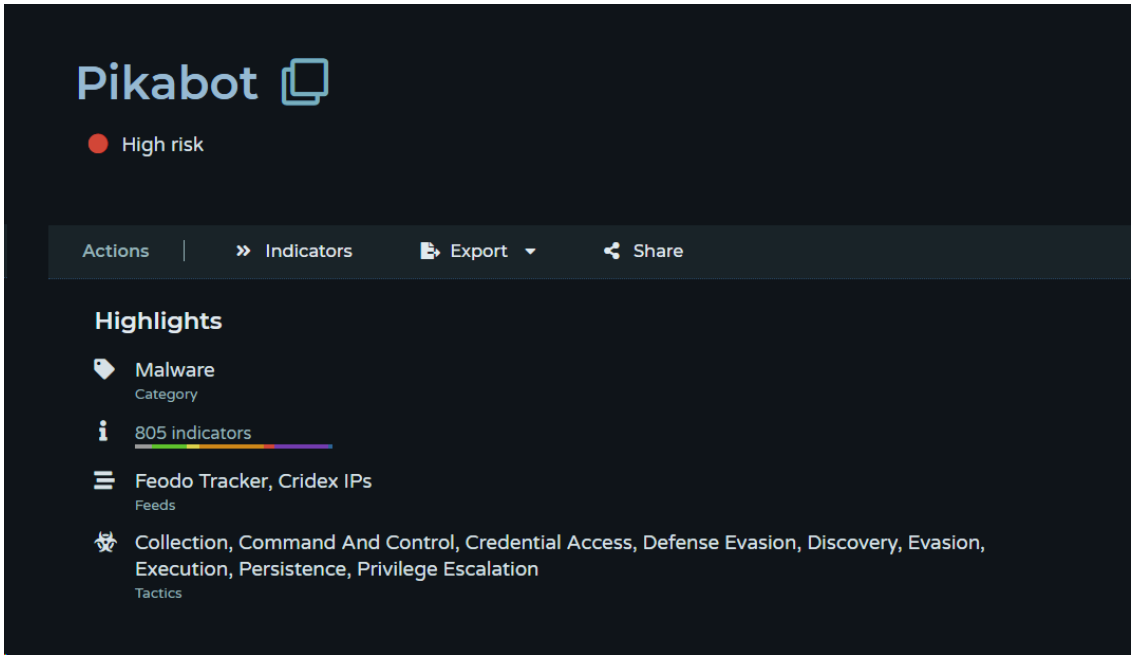


Figure 1. Pulsedive's Pikabot threat page

This blog provides an overview of the distribution mechanisms and features of Pikabot with to highlight its capabilities which include:

- Anti-analysis checks
- Loading junk libraries
- Executing commands
- Gathering system information

Distribution Mechanisms

Pikabot has been distributed primarily through spam emails. Following the takedown of Qakbot in August 2023, there was an increase in campaigns using Pikabot. Similar to campaigns spreading other malware, these emails are often part of a hijacked thread containing either an attachment or a link to download the file.

Spam Campaigns

Trend Micro Research identified a spam campaign used to deliver Pikabot. The campaign leveraged thread hijacking, where threat actors inject themselves into existing email threads and send emails as if the malicious entity was always part of the thread. Typically, Pikabot intrusions start through a JavaScript file that is used to download and execute the second stage payload. In other observed intrusions, instead of a Javascript file, the ZIP archive contained either an IMG or PDF file. In the case of an IMG, it consists of a LNK file masquerading as a Word document and the Pikabot DLL. The LNK file is used to execute the Pikabot DLL. For PDFs, the PDF serves as a lure that is used to download the DLL sample. The LOLBIN, *rundll32.exe*, is used to execute the DLL file.

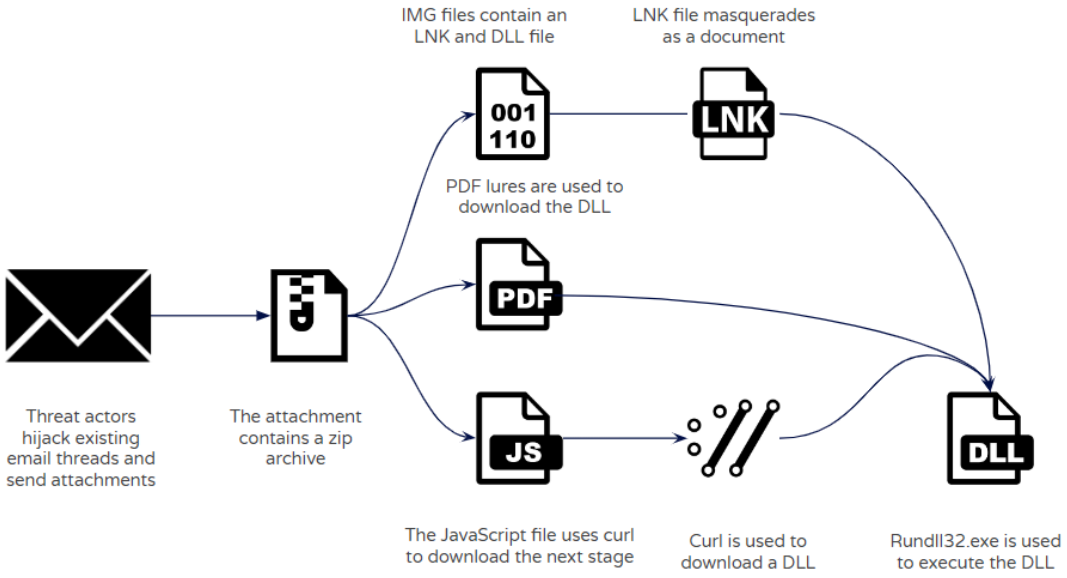


Figure 2: Execution flow observed in spam campaigns delivering Pikabot.

Malvertising

Threat researchers have also identified cases where Pikabot was distributed through malicious advertising. In one example, Pikabot impersonated the remote access tool AnyDesk in a malvertising campaign (Malwarebytes). The campaign used paid advertisements for the phishing sites to appear at the top of a user's Google search results and rely on a user clicking the sponsored link instead of the actual AnyDesk URL. After users clicked the malvertising link, they were redirected to a page mimicking the AnyDesk webpage.

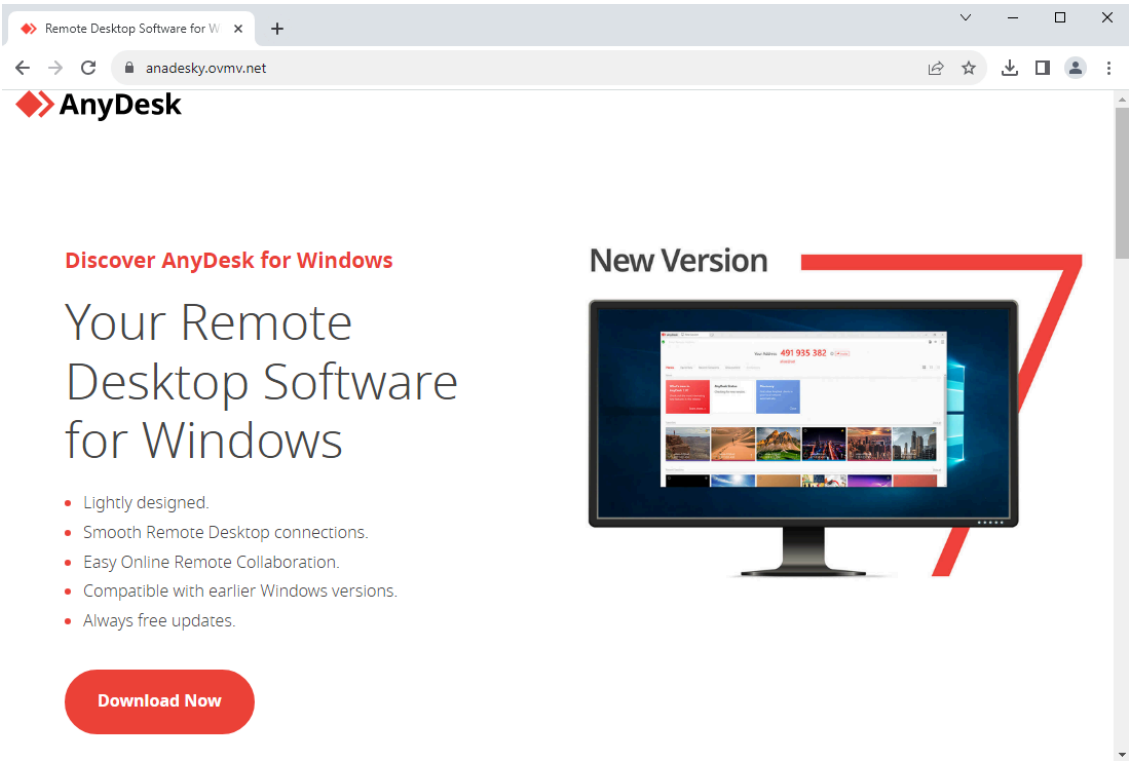


Figure 3: A web page masquerading as the official AnyDesk download source (Malwarebytes)

The page hosts a malicious .msi file used to run Pikabot. The MSI uses process hollowing to execute the malicious code in a SearchProtocolHost.exe process, by creating the process in a suspended state, unmapping the process memory, writing the malicious code to memory, and then resuming the thread.

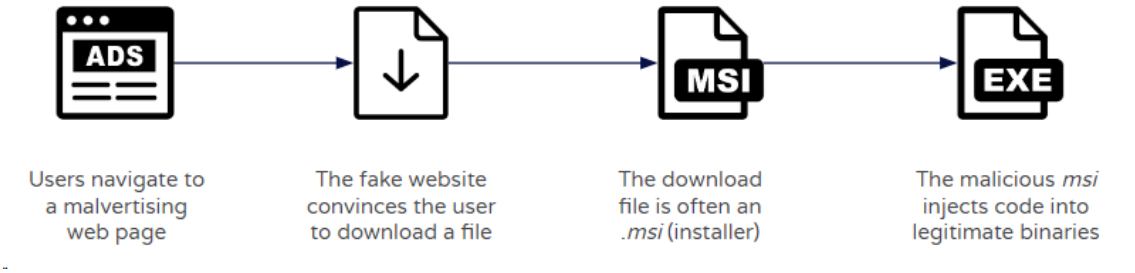


Figure 4: Execution flow observed in Pikabot intrusions originating from malvertising campaigns.

Features of Pikabot

Dynamic API Resolving

Pikabot samples resolve APIs at runtime. This means that some of the necessary APIs needed for the sample to execute are loaded during its execution and not when the program is compiled. This is an attempt by the malware author to make it more difficult to determine functionality through static analysis of imported APIs. Three functions are resolved through API hashing and once *GetProcAddress* and *LoadLibraryA* are resolved, they can be used to resolve other APIs that are stored as decrypted strings.

OALabs developed Python code that can be used to decrypt the strings. The figure below shows some of the decrypted strings from the sample 39d6f7865949ae7bb846f56bff4f62a96d7277d2872fec68c09e1227e6db9206

```
Decrypted: b'wsprintfA'
Decrypted: b'&'
Decrypted: b'&tfDgx='
Decrypted: b'whoami.exe /all'
Decrypted: b'&M1LWU='
Decrypted: b'ipconfig.exe /all'
Decrypted: b'&VC76f='
Decrypted: b'netstat.exe -aon'
Decrypted: b'&SBSlO='
Decrypted: b'{"mdPNC6f8": "%s"}'
Decrypted: b'wsprintfA'
Decrypted: b'{"mdPNC6f8": "%s"}'
Decrypted: b'wsprintfA'
Decrypted: b'HydrohemothoraxCoenaesthesis/2bQbdHQI1z9PoD?SnarlishAllobars=59eYpYysBS&UndoubtableE
Decrypted: b'&'
Decrypted: b'BaylZ'
Decrypted: b'AV89JS'
Decrypted: b'IsWow64Process'
Decrypted: b'GetProductInfo'
Decrypted: b'%d'
Decrypted: b'wsprintfW'
Decrypted: b'unknown'
Decrypted: b'GetComputerNameW'
Decrypted: b'unknown'
Decrypted: b'GetComputerNameExW'
Decrypted: b'unknown'
Decrypted: b'DsGetDcNameW'
Decrypted: b'unknown'
Decrypted: b'EnumDisplayDevicesW'
Decrypted: b'GlobalMemoryStatusEx'
Decrypted: b'GetDesktopWindow'
Decrypted: b'GetWindowRect'
Decrypted: b'%dx%d'
Decrypted: b'wsprintfW'
```

Figure 5: Python code showing the decrypted strings. Screenshot from OALabs

Anti-Analysis Techniques

The sample leverages several anti-analysis techniques to make analysis of the malware more difficult, which include:

- Using INT2D and INT3 to raise exception handlers
  - INT3 is a software breakpoint that is used to trigger an interrupt. When a debugger is not in use, the exception handler is called after an INT3 exception is raised. When a debugger is present, control of the program is not handed to the exception handler.
  - For INT2D, Windows uses the EIP register as the exception address and then increments the EIP register value. This instruction may cause issues for debuggers since increasing the EIP may cause instructions to be skipped.
- Checking the BeingDebuggedFlag in the PEB
- Calling the APIs `CheckRemoteDebuggerPresent()` and `IsDebuggerPresent()`
- Loading junk libraries
- Delaying execution using the `beep()` function

    - This is used in a similar fashion to the sleep function and is intended to delay execution. The beep function generates a noise that is played on speakers. The execution of the program waits until the sound stops to resume execution.
- Checking the value of the `NtGlobalFlag` in the process environment block to see if a debugger is attached
- Calling `NtQueryInformationProcess()` with the flag `0x7` (ProcessDebugPort)
- Using the `GetWriteWatch()` to get addresses of the allocated pages written
- Checking if the number of processors is less than or equal to 2
- Using the rdtsc instruction to detect single steps taken in a debugger
- Checking memory size with `GlobalMemoryStatusEx()` to see if it is less than 2GB

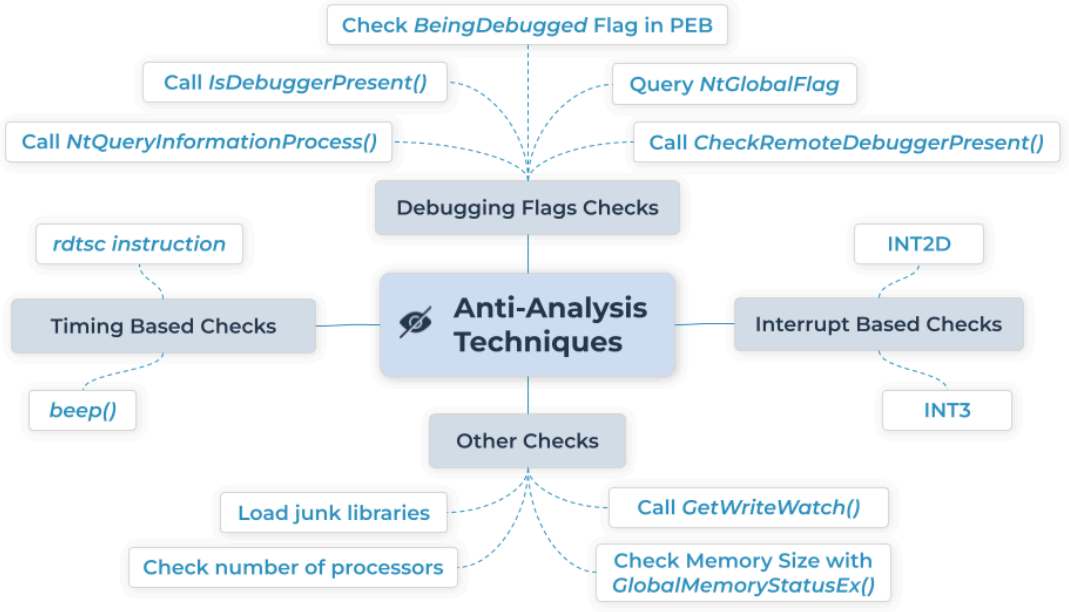


Figure 6: The anti-analysis techniques used by Pikabot

💡

Checkpoint Research

has a helpful resource covering several anti-debug tricks observed within Pikabot samples.

Language Checks

Apart from the checks outlined above which cause the program to terminate, Pikabot samples also check the language of the system to avoid infecting CIS countries. ZScaler identified that the samples check for the following languages:

- Georgian
- Kazakh
- Uzbek
- Tajik
- Russian
- Ukrainian

- Belarusian
- Slovenian

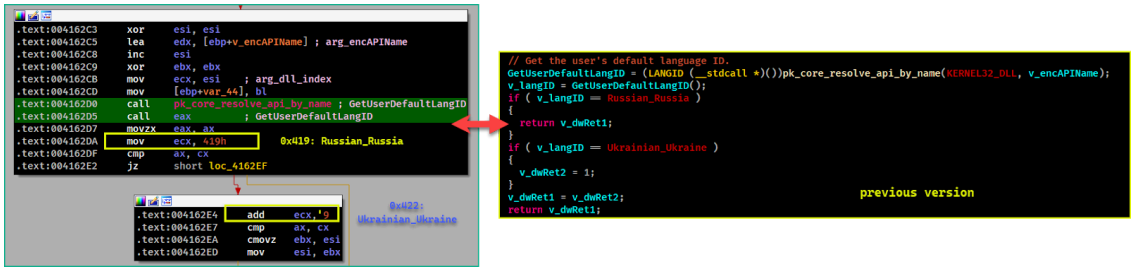


Figure 7: Pikabot uses the *GetUserDefaultLangID* function to return the language identifier of the Region set for the current user. Screenshot from 0DAY IN {REA_TEAM}.

The API function GetUserDefaultLandID returns the language for the current user. This information can also be determined from **Control Panel -> Clock, Language, and Region -> Change date, time, or number formats -> Formats** (Microsoft).

❗

Language identifiers are numerical abbreviations for languages.

If any of these languages are identified, then the program terminates. This type of check to avoid infecting endpoints in CIS countries is not unique to Pikabot and has also been observed in several ransomware variants.

Capabilities

Once the anti-analysis checks are complete, the malware initiates connections with a Command and Control server. The malware also executes several commands to gather additional information about the compromised host. The commands include collecting network information using the *ipconfig.exe* binary and collecting user information through the *whoami.exe /all* command.

Apart from using native binaries to collect system information, researchers at Zscaler identified the following capabilities that the malware can perform.

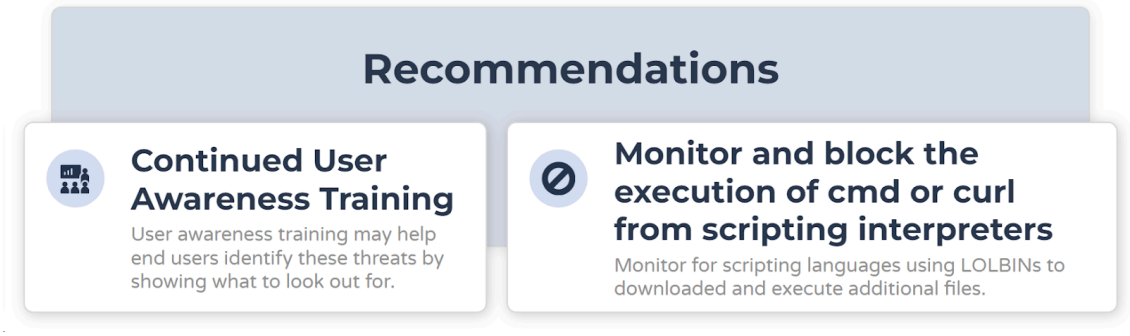
- cmd
- destroy
- shellcode
- dll
- Exe
- knock_timeout
- Information collection
    - screenshot
    - whoami
    - ipconfig
    - processes

Conclusion

Pikabot is a relatively recent loader that increased in popularity from August 2023 onwards. As of January 2024, there has been just 1 sample shared on Malware Bazaar uploaded on January 3rd, 2024. The malware adopts several anti-analysis techniques which make it difficult to detect and analyse. Pikabot binaries resolve APIs using API hashing and decrypted strings, meaning looking at imports during static analysis may not provide insight into the malware's functionality. The malware also can execute commands on the infected host, using this to conduct system information discovery activities.

Recommendations

- **Continued User Awareness Training**
  - Both of Pikabot's dissemination mechanisms require user execution. Continued user awareness training may help mitigate the risk from spam and malvertising by teaching users how these threats work and what to look out for.
- **Monitor and block the execution of cmd or curl from scripting interpreters**
  - To alert on potentially malicious VBScript or JavaScript files attempting to download additional payloads or launching recently downloaded executable content.
  - Some security products contain rules to block this type of activity. An example is the *Block JavaScript or VBScript from launching downloaded executable content* attack surface reduction rule in Microsoft Defender for Endpoint.



Indicators of Compromise

The table below contains a list of Pikabot network IoCs identified and added to the Pulsedive platform. IoCs can be queried in Pulsedive using the Explore query threat="Pikabot" and is available for export in multiple formats (CSV, STIX 2.1, JSON).

Pikabot IOCs
https://104.200.28.75:2222/

https://139.162.147.197:2225/
51.68.147.114
104.200.28.75
65.20.82.17
172.234.16.175
https://139.99.216.90:13720/nastier/YaEq5oFpdVHuvOuYK
http://65.108.216.128/l9yvUH/arcti
http://95.216.204.145/K2n/Churo
https://9jaflaverstore.com/uss/
And more, retrieve all indicators here

The table below shows some fingerprints for Pikabot C2 infrastructure:

Fingerprint Type	Fingerprint Value
JARM	21d19d00021d21d21c21d19d21d21dd188f9fdeea4d1b361be3a6ec494b2d2
JA4+	1a59268f55e5_1a59268f55e5_795797892f9c

MITRE ATT&CK TTPs

Technique	Tactic
Collection	Archive Collected Data (T1560)
	Input Capture (T1056)
Command and Control	Encrypted Channel (T1573)
	Non-Standard Port (T1571)
Credential Access	Input Capture (T1056)
Defense Evasion	Hijack Execution Flow: DLL Side-Loading (T1574.002)
	Impair Defenses: Disable or Modify Tools (T1574.002)
	Impair Defenses: File Deletion (T1070.004)
	Masquerading (T1036)
	Modify Registry (T1112)
	Native API (T1106)
	Obfuscated Files or Information (T1027)
	Process Hollowing (T1055.012)
	System Binary Proxy Execution: Regsvr32 (T1218.010)

	System Binary Proxy Execution: Rundll32 (T1218.011)
	Virtualization/Sandbox Evasion (T1497)
Discovery	File and Directory Discovery (T1083)
	Process Discovery (T1057)
	Software Discovery: Security Software Discovery (T1518)
	System Information Discovery (T1082)
	System Owner/User Discovery (T1033)
	Virtualization/Sandbox Evasion (T1497)
Evasion	Masquerading (T1036)
Execution	Native API (T1106)
	System Binary Proxy Execution: Regsvr32 (T1218.010)
	System Binary Proxy Execution: Rundll32 (T1218.011)
	Shared Modules (T1129)
Persistence	Hijack Execution Flow: DLL Side-Loading (T1574.002)

	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder (T1547.001)

Source: https://blog.pulsedive.com/pikabot/