BlackBerry Blog

BlackBerry Blog › Threat Thursday: NetWire RAT is Coming Down the Line

Threat Thursday: NetWire RAT is Coming Down the Line

RESEARCH & INTELLIGENCE / 09.16.21 / The BlackBerry Research and Intelligence Team

NetWire is a publicly available, multi-platform Remote Access Trojan (RAT) that is designed to target victims on Windows®, MacOS®, and Linux®. This threat has been distributed in phishing campaigns via weaponized Microsoft® documents, PDFs containing download links, and archive files containing payloads. It has also been seen for sale on the dark net, typically ranging in price from $40 to $140 USD.

The goal of NetWire is to perform surveillance or take control of the infected system. Once the RAT has compromised a machine, the attacker can execute a variety of remote actions from its command and control (C2) server.

The malware's surveillance abilities include logging keystrokes, capturing screenshots, and stealing passwords, as well as accessing web cameras and microphones.

Operating System

Windows	MacOS	Linux	Android
Yes	Yes	Yes	No

Risk & Impact

Risk	Medium

Technical Analysis

NetWire was first discovered in the wild in 2012, and it has been used since then by financially motivated cyber-criminals, as well as advanced persistent threat (APT) groups.

Typically, the infection vector used to distribute this malware is via phishing campaigns where victims are lured into clicking on malicious Microsoft® Office files that contain embedded macros, which then launch a payload. The malware has also been seen spreading via malicious URLs in PDF files, as well as through malicious attachments in emails.

The variant analyzed in this report was a binary created to target Windows machines. The file would arrive as a Win32 executable that is UPX-packed to help impede analysis.

The malware also uses an anti-analysis technique to avoid execution in a sandbox. The "GetCursorPos" function is called twice, as seen below, and the cursor positions are compared. The malware will not run until there is a difference in the mouse cursor positions.

Figure 1: Anti-sandbox technique

These are not the only tricks NetWire uses to hide its activity. When executed, the malicious binary creates a child process, and its code is then overwritten into this process. Because this is where the malicious activity spawns from, this makes analysis more difficult.

The parent process drops the child process into the "User/AppData/Roaming/Install" directory and exits itself. This is shown in the image below, with the filename "Host.exe."

Figure 2: Copy of NetWire in the User/AppData/Roaming/ folder

NetWire creates a registry key and adds it to the auto-run group to achieve persistence. This will ensure that the malicious file will run automatically when the victim's machine is booted up.

Figure 3: NetWire RegKey for maintaining persistence on victim's machine

By analyzing the strings in memory, we can see some of the functionality carried out by NetWire, and the information that it is attempting to gather.

It tries to gather user login data from web browsers such as Google Chrome™ and Brave® Browser.

Figure 4: NetWire browser credential-harvesting functionality

The malware also scans the directories of Microsoft® Outlook® profiles on the victim's machine to gather credentials. It also collects and logs both keystrokes and mouse movements.

The keystrokes are of particular interest to the attacker as a potential source of sensitive information that can be used for malicious activity or financial gain. This could include data such as login details for online banking sites, credit card information, corporate network access credentials, or crypto wallets. This harvested information can often fetch a hefty price on underground forums and the dark web.

NetWire creates a log file on the victim's machine in the "User/Analyst/AppData/Roaming/Logs" directory, where it stores all captured data. As shown below, the log file is named using the format DAY-MONTH-YEAR, matching the date the victim's machine was infected.

The data contained within the log file is encrypted using an RC4 cryptographic algorithm, the same algorithm the malware uses to encrypt strings and DLLs. NetWire also uses this obfuscation technique to hide registry keys, APIs, DLL names, and other strings.

Figure 6 shows the contents of the encrypted log file. The file grows larger the longer the malware remains on the machine, as it gathers more data.

Figure 6: Encrypted log file containing harvested data

The malware also creates a mutex called "OqvAvPni" on the target machine. This is used by the malware author to avoid reinfection of the same host.

Figure 7: Mutex "OqvAvPni" created by NetWire

NetWire will attempt to create a TCP connection with a remote C2 server. In the sample used in this analysis, there were continuous attempts to create a connection over port 3382 with the IP Address 192[.]169[.].69[.]25.

From the strings in memory, we can see the DNS of the C2 server that this variant is attempting to reach is "love82[.]duckdns[.]org".

Figure 8: C2 URL "love82[.]duckdns[.]org" visible in strings in memory

If this connection is successful, the attacker can transfer captured data or perform further malicious actions, such as downloading and executing additional malicious payloads.

In the instance shown below, the connection is unsuccessful as the RST flag is being returned. A TCP segment is sent with a RST flag when a connection

Figure 9: Wireshark results showing continuous TCP connection attempts with C2

YARA Rule

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
import "pe"
rule NetWire_RAT {
  meta:
    description = "Detects NetWire Remote Access Trojan"
    author = "BlackBerry Threat Research Team"
    date = "2021-09-06"
    SHA256 =
"021323e02e618769aab03cd9d5dea602ff684c2ff64ef592d69b119e78502fd9"

  strings:
    $s1 = "test.exe"
    $s2 = "key_dtor_list"
    $s3 = "new_key"
    $s4 = "./mingw-w64-crt/crt/natstart.c"
    $s5 = "./mingw-w64-crt/crt/tlssup.c"
    $s6 = "./mingw-w64-crt/crt/tlsmcrt.c"
    $s7 = "cygming-crtbegin.c"
    $s8 = "tagCOINITBASE"
    $s9 = "IID_IWinInetFileStream"
    $s10 = "winnt.h"
    $s11 = "fwrite"

  condition:
    ( uint16(0) == 0x5a4d and filesize < 1500KB and
      pe.imphash() == "084fafd5fea9d39b4ff35f2d82f0908b" and
```

```
}
```

Indicators of Compromise (IoCs)

SHA256

- 021323e02e618769aab03cd9d5dea602ff684c2ff64ef592d69b119e78502fd9

Files Created

- User/AppData/Roaming/Install/Host.exe
- User/AppData/Roaming/Logs/<Date-Of-Attack>

Domains

- love82[.]duckdns[.]org

C2

- 192[.]169[.].69[.]25

Mutex

- OqvAvPni

Registry

- HKCU\SOFTWARE\NetWire

BlackBerry Assistance

If you're battling this malware or a similar threat, you've come to the right place,
regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants
dedicated to handling response and containment services for a wide range of

clock support, if required, as well as local assistance. Please contact us here: https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

About The BlackBerry Research and Intelligence Team

The BlackBerry Research and Intelligence team is a highly experienced threat research group specializing in a wide range of cybersecurity disciplines, conducting continuous threat hunting to provide comprehensive insights into emerging threats. We analyze and address various attack vectors, leveraging our deep expertise in the cyberthreat landscape to develop proactive strategies that safeguard against adversaries.

Whether it's identifying new vulnerabilities or staying ahead of sophisticated attack tactics, we are dedicated to securing your digital assets with cutting-edge research and innovative solutions.

Corporate

Company

Newsroom

Developers

Enterprise Platform & Apps

BlackBerry QNX Developer Network

Legal

Overview

Accessibility

::: BlackBerry®

Leadership

Corporate Responsibility

Certifications

Customer Success

BlackBerry ThreatVector Blog

Developers Blog

Help Blog

Privacy Policy