# **Malware Analysis Report: SMB.exe/SMB.dll**

## **Overview**

This report provides an analysis of a suspected malware sample. The first section covers included DLLs. DLLs are not typically executed in a linear or sequential manner; instead, they expose callable functions and classes that may be invoked by an external process in a non-linear, event-driven, or conditional fashion. Accordingly, this analysis focuses on **logically significant components** of the DLL, rather than attempting a start-to-finish execution trace.

All DLL files in this sample are .NET assemblies developed in C#.

**Note:** For security reasons, code excerpts have been reduced to **syntaxless pseudocode**, and certain portions of the code have been redacted.

**UPDATE AS OF 8/17/25: THIS MALWARE SAMPLE IS NOW INERT! THE C2 DOMAIN HAS BEEN TAKEN DOWN AFTER MY REPORTS, RENDERING THE MALICIOUS SECTIONS OF CODE INERT**

## **Namespace 1: DnsResolver.Services**

### **Class: Utils**

| **Function** | **Description** |
| --- | --- |
| GetHostName() | Returns the local machine's hostname. |
| GetDomainName() | Queries Win32\_ComputerSystem via WMI to determine if the device is domain-joined; returns the domain name or a status message. |
| GetLocalIPAddresses() | Returns a list of all local IPv4 addresses on the system as strings. |
| createTaskId(ip) | Generates a numeric ID from an IP address by summing its octets (e.g., 192.168.0.10 → 370). |

**Notes:**

* All methods in this namespace are callable from any external process.

## **Namespace 2: SMB**

Code is heavily redacted for security purposes.

### **Key Functionality**

#### **Agent ID Management**

FUNCTION getAgentId()

IF AgentId is null THEN

IF file at agentIdFilePath exists THEN

READ contents of file

SET AgentId = file contents

ELSE

GENERATE new 5-character ID using IdGenerator

SET AgentId = generated ID

WRITE AgentId to agentIdFilePath

END IF

END IF

RETURN AgentId

* agentIdFilePath = cfg.dll
* cfg.dll stores a 5-character ASCII string generated by IdGenerator.
* **Agent ID role:** Critical for victim authentication to the C2 server. A new ID is only generated if the file is missing. Modifying the file does **not** trigger a new ID.

#### **Victim Authentication Flow**

IF HttpHelper.Get(Http.BaseURL + "register/?s=1", "token", getAgentId()) == "ok"

RETURN true

ELSE

RETURN false

* Http.BaseURL points to the C2 beacon (defanged: hxxp://api.fastrdp.online/api/).

#### **Additional Functionality**

* Remote command execution via headless PowerShell.
* Reports system uptime to C2.
* Reads and modifies various Windows registry keys.

## **Dynamic Analysis Findings**

* Conditional logic appears to filter based on:  
  + Domain membership.
  + Device uptime
  + Hardware specifications (CPU, RAM).
* Confirms client authentication workflow and C2 registration.
* Remote command execution is confirmed to be possible; may serve as a dropper.
* Exfiltration occurs upon program launch:  
  + Heartbeat sent to API over HTTP.
  + Data is sent to C2 and appears fragmented and encrypted across TCP segments.
* Reads a wide array of registry keys.
* All program modules have **timestomped metadata**.
* Modifies onedrive files, unable to analyze further due to sample being inert (spoken of earlier in paper).
* Observed C2 endpoint IPv4 address: 64.44.154.197 (hosted by Nexeon Technologies as of 08/06/25).

## **Summary**

The SMB-release sample exhibits standard malware behaviors:

* Victim authentication and fingerprinting.
* Remote command execution and dropper capabilities via headless powershell.
* Data exfiltration over encrypted TCP, possibly living off the land with bcrypt.
* Anti-forensic techniques such as timestomping of process modules.
* Environmental awareness through checks for domain membership, possibly virtualization, uptime, Win Defender configuration, and hardware specs.
* Seems to not be persistent by default, but could be if the C2 operator used the dropper feature spoken of earlier.

This analysis provides both static and dynamic insights into the sample, highlighting its C2 communications.

## **Removal and Mitigation**

The following section contains information related to the removal of the malware from an affected machine.

* After discovery of compromise by this malware, the infected machine should be **IMMEDIATELY** disconnected from all networks.
* Full disk format/secure erase all disks connected to the infected device, re-install operating system from a trusted source. **DO NOT DO AN IN PLACE UPGRADE/REINSTALL.**
* If other removal tips can not be done, modify the hosts file so that any DNS queries for the malware’s C2 domain resolve to an inert address.
* If the C2 domain is no longer registered/active, the malicious code will not execute successfully. (no longer registered as of 8/17/25)

## 

## 

## 

## 

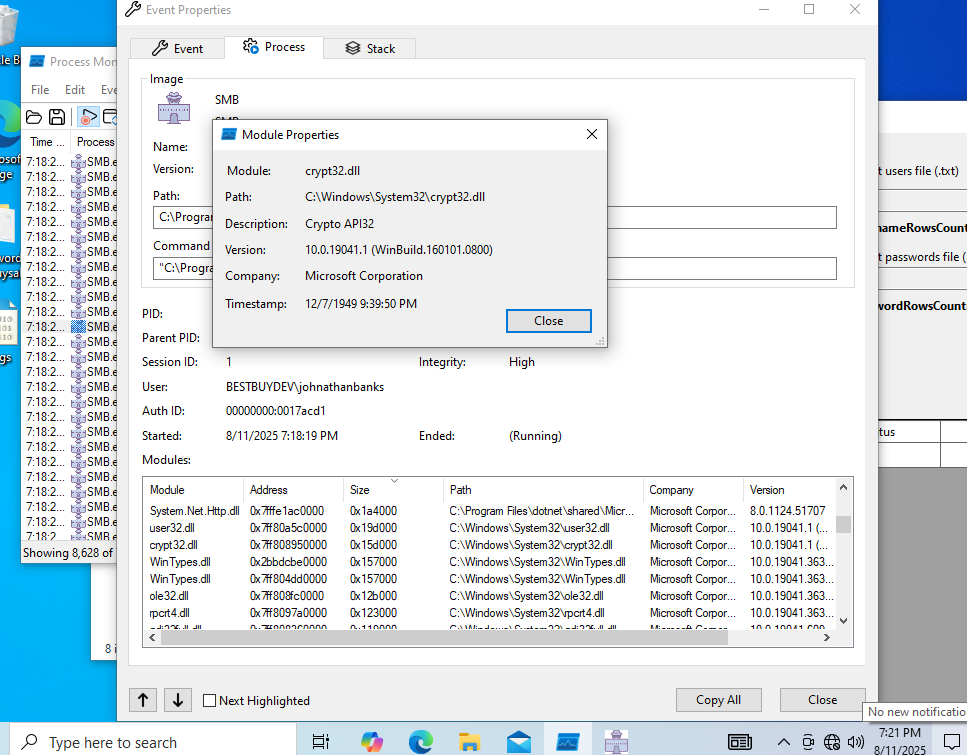
## 

## 

## 

## 

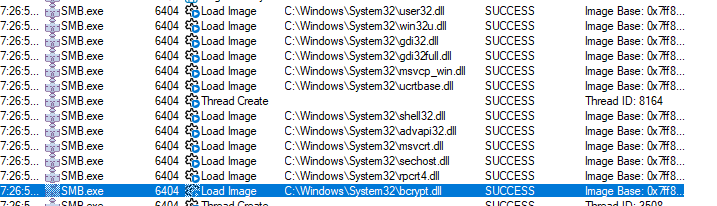
## **Images**



*Timestomping*

**

*Agentid generation snippet*

**

*Use of bcrypt.dll and other important .dll files*

**

*Anti debugging instructions*

**Hashes**

* SHA256 9B9689ADEF78ADF5AEA081699F229F5189DD90BDA174348DF5BF6FB36B91F852 (fastSMB.zip)
* SHA256 EF9E28020A5111E96A51BA41590D6F7586E639108CDD18262548216C0F605B9E (smb.dll)
* SHA256 48A2F1A71ED64B59625AF67C404EBB7414C5B095A3AACB45E9B962342C211FAA (smb.exe)