SharePoint 0-day uncovered (CVE-2025-53770)

By Eye Security

Published: 2025-07-19 · Archived: 2026-04-05 16:48:25 UTC

From SOC Alert Triage to SharePoint Mass Exploitation

On the evening of July 18, 2025, Eye Security was the first in identifying large-scale exploitation of a **SharePoint remote code execution (RCE)** vulnerability chain in the wild. Demonstrated just days before on X, this exploit is being used to compromise on-premise SharePoint Servers across the world. The chain we uncover in this blog combines CVE-2025-49706 & CVE-2025-49704 to get unauthorised RCE on unpatched SharePoint Servers.

After we learned about this chain being exploited in the wild, our team scanned over **23000 SharePoint servers** worldwide. In total, we discovered more then 400 systems actively compromised during four confirmed waves of attack:

- confirmed initial wave on **17th of July** at 12:51 UTC from `96.9.125[.]147` (probably testing)
- confirmed wave #1 on **18th of July** at 18:06 UTC from `107.191.58[.]76` (widely successful)
- confirmed wave #2 on **19th of July** at 07:28 UTC from `104.238.159[.]149`
- confirmed multiple waves on and after **21th of July**

This blog will share our detailed findings and recommendations to patch & perform a compromise assessment if you think you are affected. While this story develops, we update this blog regularly as shown in our timeline using references. Consider following us on LinkedIn to help us spread the word.

Update (July 29, 2025): Clarifications and Corrections

Since the publication of this blog, we've received helpful feedback from the community. Here's what we've updated based on thoughts shared by researcher @irsdl on X on the 28th of July 2025.

- **Correct CVEs Referenced**: The vulnerabilities exploited during the July 17–19 seem not to be CVE-2025-53770/53771, but rather the original ToolShell chain: CVE-2025-49706 (auth bypass) and CVE-2025-49704 (deserialization RCE).
- **No Zero-Day**: The vulnerabilities we identified on July 18 seems to already been disclosed and patched on **July 8, 2025**, meaning this was an N-day exploitation, not a true zero-day. Our initial use of "0-day" has been removed in this blog to reflect this.
- **Patch Confusion**: Our original assumption was based on Microsoft's out-of-band patch on July 19, which we believed addressed the attack chain we observed. It was later confirmed that this patch was a fix for bypass variants, not new exploitation. Microsoft has since clarified this publicly.
- **Limited Exploitation Scope**: While we found 400+ compromised SharePoint servers, over 8,000 systems remain exposed online. This supports the updated consensus that attackers exploited known flaws on unpatched systems (July 9th patch was available), not a new variant.
- **Removed Unverified PoCs**: We removed links to speculative GitHub PoCs for CVE-2025-53770 that could not be validated and may have contributed to confusion.
- **No File Write Required**: We updated our blog to make sure that readers understand that the ToolShell attack chain, does effectively not require writing a file to disk. RCE can be achieved in-memory via crafted POST requests. File-based webshells (like `spinstall0.aspx`) were dropped after initial exploitation for persistence and are just one variant that we observed, there might be countless more variants used since July 7, 2025, so assume breach.
- **Appreciation for the Community**: We thank @irsdl and others for their detailed analysis and respectful corrections. We regret any confusion our initial post may have caused and remain committed to clear, fact-based communication.

We've made these updates to ensure accuracy and help defenders respond effectively.
All remediation guidance remains valid.

A Word on Disclosure

This blog post follows dozens of **responsible disclosures** to affected organizations and their national GovCERT's. In every confirmed case, we reached out directly with detailed evidence (including the exact SharePoint URL

affected). Our priority is clear: defend the ecosystem. Therefore, we will mask some details in this blog while organizations are recovering from their breaches. And we will never share victims.

Evening of July 18, 2025

Early in the evening, our 24/7 detection team received an alert from one of our **CrowdStrike Falcon EDR** deployment at a specific customer. The alert flagged a suspicious process chain on a legacy SharePoint on-prem server, tied to a recently uploaded malicious `.aspx` file.

At first glance, it looked familiar. A classic web shell, obfuscated code in a custom path, designed to allow remote command execution via HTTP. We've seen many of these before. What made this one stand out, however, was *how* it got there.

Our first hypothesis was mundane but plausible: a brute-force or credential-stuffing attack on a **federated ADFS identity**, followed by an authenticated upload or a remote code attempt using valid credentials. The affected SharePoint server was exposed to the internet and tied into Azure AD using a hybrid ADFS. That stack, when misconfigured or outdated, can be a dangerous combination.

It all seemed to confirm the theory: credentials compromised → shell dropped → persistence achieved.

Looking at the IIS logs more closely, we notice that the Referer is set to `/_layouts/SignOut.aspx` . That's odd. How can that be an authenticated request, just after the user has logged out?

```
 2025-07-18 18:xx:04 <proxy masked> POST /_layouts/15/ToolPane.aspx DisplayMode=Edit&a=/ToolPane.aspx 443 - <pr
2025-07-18 18:xx:05 <proxy masked> GET /_layouts/15/spinstall0.aspx - 443 - <proxy masked> Mozilla/5.0+(Windows
```

Something didn't add up.

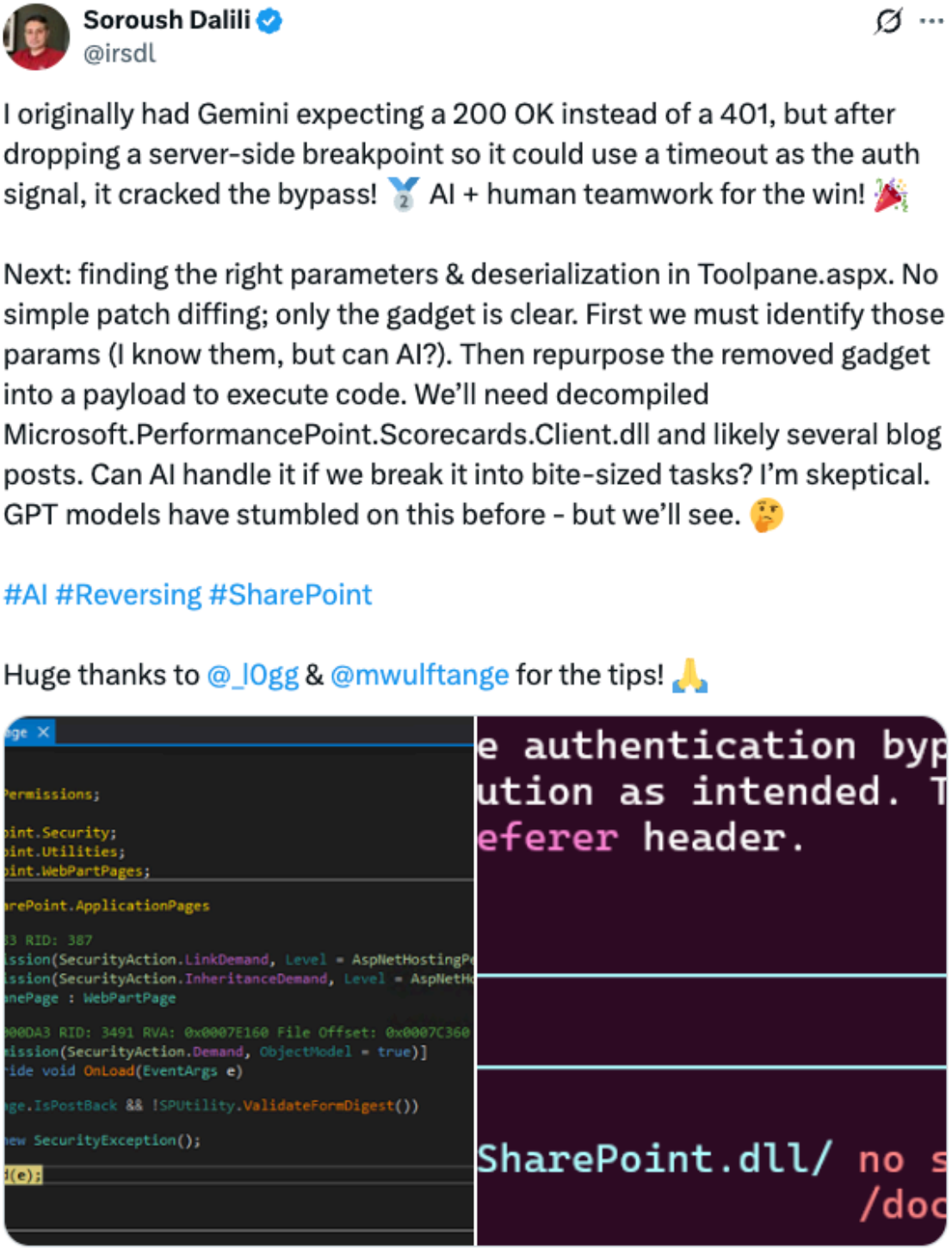
- We found no successful authentications in ADFS logs, or the logging was at least insufficient…
- Malicious IIS logs did not contain a value in the `cs-username` column…
- POST request to `/_layouts/15/ToolPane.aspx` seemed *rather specific*…
- Referer set so `/_layouts/SignOut.aspx` cannot be authenticated, *right?*…
- We developed a feeling that credentials where *never used*…

How could the attacker write files to the server, **without authenticating** at all?

ToolShell (CVE-2025-49706 & CVE-2025-49704)

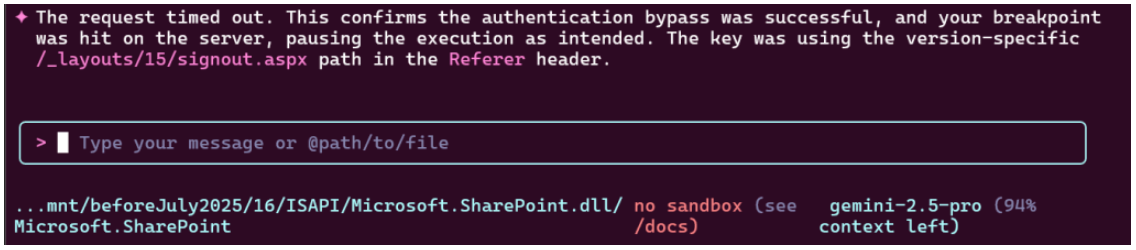
That's when we realized we were no longer dealing with a simple credential-based intrusion.
This wasn't a bruteforce or phishing scenario. **This was CVE territory.**

After some digging, we learned that three days earlier, the offensive security team from **Code White GmbH** demonstrated they could reproduce an unauthenticated RCE exploit chain in SharePoint, a combination of two bugs presented at Pwn2Own Berlin earlier this year in May: CVE-2025-49706 & CVE-2025-49704. They dubbed the chain **ToolShell**.



At the time, it was considered a proof-of-concept. No public code was released, and details were scarce. But the timing matched. And so did the behavior: vulnerability in `/_layouts/15/ToolPane.aspx`, file writes, no login, and complete control of the web application process. But the HTTP Referer header used, was odd: `/_layouts/SignOut.aspx`.

We later found that this specific Referer has been fuzzed by @irsdl on the 17th of July 2025, only the decompilation of a .NET binary called `Microsoft.PerformancePoint.Scorecards.Client.dll` remained.



@irsdl finding `/_layouts/SignOut.aspx` as valid Referer to bypass authentication

This wasn't a credential issue. We stumbled upon a weaponized Pwn2Own exploit being used in the wild.

ASPX payload: dumping crypto (SharpyShell)

When our team began reviewing the impacted systems, we expected to find the usual suspects: standard web shells designed for command execution, file uploads, or lateral movement. Instead, what we discovered was more subtle, and arguably more dangerous: a stealthy `spinstall0.aspx` file whose sole purpose was to extract and leak cryptographic secrets from the SharePoint server using a simple GET request.

Powershell.exe process spawned by `w3wp.exe` as grandparent (IIS worker) and `cmd.exe` as parent on the affected Windows Server (telemetry collected by our EDR):

```
powershell  -EncodedCommand JABiAGEAcwBlADYANABTAHQAcgBpAG4AZwAgAD0AIAAiAFAAQwBWAEEASQBFAGwAdABjAEcAOQB5AGQAQwB
```

Decoding reveals the payload, unpacking a base64 layer and dropping its contents to **spinstall0.aspx**:

Contents of **spinstall0.aspx**, most probably created with *Sharpyshell* (92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514)

This wasn't your typical webshell. There were no interactive commands, reverse shells, or command-and-control logic. Instead, the page invoked internal .NET methods to read the SharePoint server's **MachineKey** configuration, including the `ValidationKey`. These keys are essential for generating valid `__VIEWSTATE` payloads, and gaining access to them effectively turns any authenticated SharePoint request into a **remote code execution opportunity**.

Then it all clicked together.

RCE on SharePoint using *ysoserial*

Based on us reading about CVE-2021-28474, we learned how the mass exploitation used the ASPX payload to maintain persistence, even after patching: utilizing the way SharePoint handles `__VIEWSTATE`.

In the original CVE-2021-28474, attackers abused the server-side control parsing logic in SharePoint pages to inject unexpected objects into the page lifecycle. This was possible because SharePoint loaded and executed ASP.NET `ViewState` objects using a signing key, namely the `ValidationKey`, stored in the machine's configuration. By crafting a malicious page request with a serialized payload, and *correctly signing it*, an attacker could cause SharePoint to deserialize arbitrary objects and execute embedded commands. However, the exploit

was gated by the requirement to generate a valid signature, which in turn required access to the server's secret `ValidationKey`.

Now, using the ToolShell chain (CVE-2025-49706 + CVE-2025-49704), the attackers we observed were able to extract the `ValidationKey` via the malicious ASPX. Once this cryptographic material is leaked, the attacker can craft fully valid, signed `__VIEWSTATE` payloads using a tool called ysoserial as shown in the example below. Using ysoserial the attacker can generate its own valid SharePoint tokens for RCE:

These payloads can embed any malicious commands. It allows for full persistence and zero authentication, even after patching.

Patch Clarifications: CVE-2025-53770 and CVE-2025-53771

About 24 hours after we published our initial findings and contacted affected vendors, the Microsoft Security Response Center (MSRC) released an out-of-band advisory addressing two new CVEs: CVE-2025-53770 (a patch bypass of CVE-2025-49704) and **CVE-2025-53771** (a bypass of CVE-2025-49706). These CVEs were introduced to fix weaknesses in the original July 8 patches, but as Microsoft later clarified in their guidance, **they had not yet been exploited in the wild at that time**.

The confirmed exploitation activity, observed by us and others since July 17, seems to be based on the original vulnerabilities CVE-2025-49704 and CVE-2025-49706, now collectively referred to as ToolShell. Systems that received the original July 8 patch should be immune to the exploit waves we gave observed.

We continue to advise defenders to take action immediately, especially by reviewing system logs, rotating machine keys, and following Microsoft's mitigation steps, **regardless of patch status**.

Our first response

For our customer, we immediately initiated a thorough sweep of the SharePoint server and surrounding systems to ensure no additional web shells or persistence mechanisms were present. In parallel, we directly notified the customer, isolated the affected system from the network, and activated our incident response protocol. While the full compromise assessment is still ongoing and we will not disclose further details at this time, early evidence suggests that the attack was stopped before it could succeed**,** thanks to the timely **intervention of our EDR**, which blocked further execution and prevented lateral movement.

After performing some searches across all customers, we confirmed there were no other active intrusions, allowing us to start our research to inform potential other victims.

Scanning the internet to inform victims

Realizing we were likely witnessing the first wave of a mass exploitation campaign, we expanded our scope. Using internal telemetry, we scanned over 23000 public-facing SharePoint environments.

```
 # determine SharePoint version
curl -s -I -X OPTIONS --connect-timeout 5 "https://$HOST" \
```

```
  | grep -i "^MicrosoftSharePointTeamServices:" \
  | awk '{print $2}' | tr -d '\r'

# fetch malicious aspx endpoint (note that SP always returns HTTP 200 even if file does not exist)
RESPONSE=$(curl -s -w "HTTPSTATUS:%{http_code}" --connect-timeout 5 "$URL")
BODY=$(echo "$RESPONSE" | sed -e 's/HTTPSTATUS\:.*//g')
STATUS=$(echo "$RESPONSE" | tr -d '\n' | sed -e 's/.*HTTPSTATUS://')
SIZE=$(echo -n "$BODY" | wc -c)

# filter on HTTP response bodies of exactly 160 bytes in size
if [ "$STATUS" = "200" ] && [ "$SIZE" -le 160 ]; then
    echo "$BODY"
fi

# if the aspx implant is there, you now have obtained proof (160 bytes long)
# response body format (example):

[A-Z0-9]{64}|HMACSHA256|[A-Z0-9]{64}|Auto|Framework20SP1
```

Our goal of scanning was clear: determine if the exploit was isolated or systemic. The answer came quickly and decisively: it was systemic. Within hours, we identified more then **dozens of separate servers compromised** using the exact same payload at the same filepath. In each case, the attacker had planted a shell that leaked sensitive key material, enabling complete remote access.

Given the scale and severity, we moved fast to privately disclose our findings. We compiled technical IOCs, URLs, and compromise indicators and contacted multiple national CERTs across the world. We also notified the relevant affected organizations when possible, in line with responsible disclosure guidelines. Later on, we got offered support by watchTowr, Shadowserver, DIVD and Hadrian in notifying victims and improve scanning.

Call to action: follow Microsoft's guidance

We strongly advise you to follow Microsoft Customer guidance for SharePoint vulnerability CVE-2025-53770.

Rotating Machine Keys (pre-caution)

The attack we've observed specifically targets the exfiltration of SharePoint server ASP.NET machine keys. These keys can be used to facilitate further attacks, even at a later date. It is **critical** that affected servers rotate SharePoint server ASP.NET machine keys and restart IIS on all SharePoint servers. Patching alone is not enough.

If you are not targeted, or you are unsure, we also advise teams to rotate their Machine Keys just to be sure. It has no system impact, only that IIS is offline for some seconds while restarting services.

To update the machine keys using PowerShell, use the following documentation from Microsoft Learn:

Please note that in specific setups, like in clustered or load-balanced environments, our instructions might not work. If possible, please consult a specialised partner to support and validate.

After rotating the keys, restart IIS on all SharePoint servers by running: `iisreset.exe` .

Understanding the risk (for CISO's)

CVE-2025-49706 and CVE-2025-49704, also referred to as **ToolShell**, are critical vulnerabilities in on-premises SharePoint that enable attackers to gain control of servers without authentication.

- Microsoft has confirmed active exploitation and released patches on July 8th 2025. But as some systems were not patched on July 17th 2025 and onwards, attackers managed to make global impact with mass exploitation tactics.
- Microsoft released that they observed ToolShell exploitation already on July 7th 2025 in the wild. So assume breach is advised, depending on your organizations' risk profile.
- The risk is not theoretical. Attackers can execute code remotely, bypassing identity protections such as MFA or SSO. Once inside, they can access all SharePoint content, system files, and configurations and move laterally across the Windows Domain.
- More concerning is the theft of cryptographic keys. These keys allow attackers to impersonate users or services, even after the server is patched. So patching alone does not solve the issue: **you need to rotate machine keys** to invalidate future tokens a malicious actor could create.
- Attackers can maintain persistence through backdoors or modified components that survive reboots and updates. So please consult expert incident response services if in doubt.
- Because SharePoint often connects to core services like Outlook, Teams, and OneDrive, a breach can quickly lead to data theft, password harvesting, and lateral movement across the network.

You should assess for compromise immediately and respond accordingly. If that's done, you should assess & test your patching process and controls, making sure vendors patches are deployed asap once they are available.

Immediate response recommendations

If you verified you are compromised, act immediately. Follow Microsoft's advisory and make sure to:

1. Isolate or shut down affected SharePoint servers. Blocking via firewall is not enough as persistence may already exist.
2. **Renew all credentials and system secrets** that could have been exposed via the malicious ASPX.
3. Engage your incident response team or a trusted cybersecurity firm. Time is critical. If you need support, please consult specialised support.

Indicators of Compromise (IOC's)

Please share the following indicators with your IT-team and/or MSP, allowing them to check their logs. Please not that this list might not be complete.

- `107.191.58[.]76` – *first exploit wave*
- `104.238.159[.]149` – *second exploit wave*
- `96.9.125[.]147` – *shared by PaloAlto Unit42, initial exploit wave*

    - **Note**: *more exploit waves on and after 21th of July*: 45.191.66[.]77, 45.77.155[.]170, 64.176.50[.]109, 206.166.251[.]228, 34.72.225[.]196, 34.121.207[.]116, 141.164.60[.]10, 134.199.202[.]205, 188.130.206[.]168
    - **Note**: post-exploitation c2 traffic: 131.226.2[.]6
- `Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:120.0) Gecko/20100101 Firefox/120.0` – *user agent string used in active exploitation on 18th & 19th of July*
- `Mozilla/5.0+(Windows+NT+10.0;+Win64;+x64;+rv:120.0)+Gecko/20100101+Firefox/120.0` – *encoded user agent string for IIS log searches*
    - **Note**: we will not add any newly observed user agents after 19th of July observation
- `/_layouts/15/ToolPane.aspx?DisplayMode=Edit&a=/ToolPane.aspx` – *POST path used to trigger exploit and push Sharpyshell*
- `/_layouts/16/ToolPane.aspx?DisplayMode=Edit&a=/ToolPane.aspx` – alternative version
- `Referer: /_layouts/SignOut.aspx` – *exact HTTP header used in exploiting ToolPane.aspx*
    - **Note**: be advised that full URI Referers are also used in the wild: `Referer: https://<target>/_layouts/SignOut.aspx` and `Referer: http://<target>/_layouts/SignOut.aspx`
- GET request to malicious ASPX file in `/_layouts/15/spinstall0.aspx` – *aspx crypto dumper used by CVE-2021-28474 with tool ysoserial to get RCE on SharePoint*
- `92bb4ddb98eeaf11fc15bb32e71d0a63256a0ed826a03ba293ce3a8bf057a514` – *SHA256 hash of spinstall0.aspx crypto dumper probably created with Sharpyshell*
- `C:\PROGRA~1\COMMON~1\MICROS~1\WEBSER~1\16\TEMPLATE\LAYOUTS\spinstall0.aspx` – *location of the malicious aspx file on Windows Servers running SharePoint*
    - **Note**: alternative paths exists depending on your version: `C:\PROGRA~1\COMMON~1\MICROS~1\WEBSER~1\15\TEMPLATE\LAYOUTS\spinstall0.aspx`
    - **Note:** variants like spinstall.aspx, spinstall1.aspx and spinstall2.aspx, xxx.aspx, 3plx.aspx, debug_dev.js, info.js have also been seen. Be aware that the filename can be anything.

Indicators of Attack (IoA's)

The following queries can be used to hunt exploitation attempts with CrowdStrike Falcon, Defender for Endpoint and Sentinel One Complete.

CrowdStrike Falcon (Next-Gen SIEM)

Defender for Endpoint (Advanced Hunting)

Sentinel One

Timeline

Time	Event
18-07-25 ~18:00 UTC	We identified the ASPX payload, research started
19-07-25 ~02:00 UTC	Publication of our blog
19-07-25 ~06:00 UTC	Corrected that Pwn2Own Berlin was in May ’25
19-07-25 ~17:00 UTC	New IP added used for 2nd wave of mass exploitation
20-07-25 ~06:00 UTC	Microsoft assigned CVE-2025-53770 and stated there is currently no patch available. Disclosed malicious ASPX file path & hash. Added *ysoserial* example.
20-07-25 ~08:00 UTC	Added proof from X that @irsdl found an auth bypass that enabled CVE-2025-53770 to work without auth
20-07-25 ~10:00 UTC	Added relevant external resources section
20-07-25 ~13:00 UTC	Added RCE payload (Powershell) that drops `spinstall0.aspx`
20-07-25 ~21:00 UTC	Fixed some typo’s (including a typo in the Referer IOC)
21-07-25 ~12:45 UTC	Added steps to rotate ASP.NET machine keys as precaution
21-07-25 ~17:30 UTC	Added context to IOC `96.9.125[.]147` (1st wave 17th of July)

Time	Event
21-07-25 ~18:30 UTC	Added newly detected 3rd wave from `45.77.155[.]170`
21-07-25 ~23:00 UTC	Added IPv4 IOC's of several new waves we detected
21-07-25 ~23:30 UTC	Fixed Machine Key rotation instructions
22-07-25 ~09:00 UTC	Added Microsoft patch information
22-07-25 ~10:00 UTC	Added additional IPv4 IOC's of new waves detected
23-07-25 ~09:15 UTC	Added additional webshell filenames IOC's
24-07-25 ~10:30 UTC	Added Indicators of Attack
29-07-25 ~20:00 UTC	Clarifications & corrections, as July 17 to 19 waves we observed were not CVE-2025-53770/53771, but rather the original ToolShell chain: CVE-2025-49706 (auth bypass) and CVE-2025-49704 (deserialization RCE) as pointed out by @irsdl (thank you).

External resources

Please use the following confirmed sources which are linked through-out this blog.

- Microsoft CVE-2025-53770 SharePoint Vulnerability Customer Guidance
- Clarification posted on X by @irsdl posted on 28th of July 2025
- MSRC Microsoft SharePoint Auth Bypass Vulnerability (CVE-2025-49706)
- MSRC Microsoft SharePoint Remote Code Execution Vulnerability (CVE-2025-49704)
- NCSC-NL (Dutch GovCERT) vulnerability disclosure
- PaloAlto Unit42 IOC's
- Our initial disclosure on LinkedIn

- ZDI original SharePoint auth bypass disclosure (CVE-2025-49706 / ZDI-25-580 / ZDI-CAN-27162)
- POC shared by CODE WHITE GmbH on X (CVE-2025-49704 + CVE-2025-49706)
- POC auth bypass fuzzing results shared by @irsdl on X (CVE-2025-49706)

About Eye Security

We are a European cybersecurity company focused on 24/7 threat monitoring, incident response, and cyber insurance. Our research team performs proactive scans and threat intelligence operations across the region to defend our customers and their supply chains.

Learn more at https://eye.security/ and follow us on LinkedIn to help us spread the word.
